

代码能有多难？

学习？噢不，我们只是聊天

目錄

前言

也没有什么高大上的想法，就是想把自己会的一点网页代码的知识拿出来跟大家聊聊。你没看错，就是聊聊，就像我们平时在夜幕降临之后，坐在路边的小摊里撸串的感觉，没有玄奥的理论，没有教条的讲解，咱们就是聊天。

本书适合于零基础的小白们，高手请飞过，因为在你看起来本书行文会显得比较的拖沓，当然你拿它当段子三百首看我也是十分开心的。然后就是那些太白的小白也爬过吧，虽然代码很简单，我讲得也很细致，但是我还是不能从切换中英文输入法讲起的，换言之，你总是要懂得日常的电脑操作方法我们才有交流的基础。希望这个要求你不会觉得太高。

然后说一个很伤感情的问题，本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com**（刘源）

所有付费的交易号我都会存留，将来有各种活动可以享受优先权，总之各种福利。

QQ 群：318988305（为了简单粗暴地防止广告，所以入群收费一块钱）

大家可以在里面自助解答问题，由于时间关系，我无法承诺解答所有问题，但是我会尽力。

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发： http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书： http://www.jianshu.com/users/d37eaf3de0ff
- 站酷： http://www.zcool.com.cn/u/12927742

第一章 开场小谈

写教程的想法由来已久了，因为，当初我也是一只小白，其实谁也不是一开始就会的，只不过闻道有先后罢了。遥想我还很白很白的时候啊，那时候白的连博客和论坛都分不清楚，你也别指望我会去通过搜索解决问题，而且仿佛那会也没有那么多解决问题的资料。好吧，应该说是我不会找，因为问题总是应该出在自己身上的。

总而言之吧，想学点东西，想弄懂点问题真的挺难的。遇到一个高人，毕恭毕敬，诚惶诚恐，一肚子问题都不敢多问，怕给人家问烦了不理我。但是值得庆幸的是，我遇到的高手大多耐心非常，一遍两遍三遍四遍的给我讲，然后我还没搞明白，他们又手把手的给我演示……现在想起来依旧为他们的热情和耐心感动不已。

说知恩图报有点严重了，可是被人家这么感动过之后就总想着做点什么，但是那些高手我真的帮不上人家，那么便转过头来把我会的东西讲给后来的人吧，算是一种传承，一种接力，或者只为心安。我会努力的在后面的内容中把各种知识点给大家讲的明白，讲的容易记住，也希望如果大家在本书中学到了什么的话，那么在将来有小白向你发问的时候，你能耐心的为他解答一二。

当然啦，其实你们现在的学习环境比我们那会好多啦，毕竟教程网站层出不穷的，剩下只是想不想学的问题了。然后你恨恨的咬了一口烤串跟我说——想学啊！但是每次看到第三章就看不懂了。得，别提这糟心的事，咱们继续扯淡。

> 本章学习卡片：卡片 1

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com**（刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发：http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书：http://www.jianshu.com/users/d37eaf3de0ff
- 站酷：http://www.zcool.com.cn/u/12927742

第二章 扯扯学习方法

你说，呵呵，从小学三年级老师就一直念叨这个，现在你又扯这个，有意思么？

要是我跟他们说的一样肯定没意思啊，但是要是不一样呢？

1. 不求弄懂，但要坚持；
2. 不求记住，但要勤用；
3. 不求理论，但要效果；
4. 多去实践，然后搜索，最后才是提问。

嘿嘿，我这要求够低了吧？跟老师那个考试要考、考试不考的学习方法大不一样哦～现在我来说说都是啥意思：

一、不求弄懂，很多新手上来犯的第一个错误就是：总想把今天学的东西弄懂了。我跟你说啊，这事不大可能，就好像说你现在很饿，我给了你两粒旺仔小馒头，你吃了，没啥效果，你就因此断言，这东西不可能吃饱！这挺搞笑的吧，你吃五斤试试。现在的问题是，你刚学了一点点（吃了一两个小馒头），就问怎么搞不懂（为啥吃了没感觉）？这不是搞笑么，想学懂（吃饱）继续学（吃）就是了。当然了，因为这个原因而半途而废也是挺可惜的，所以说要坚持，这个跟毅力无关啊，只是你没吃饱呢就继续吃，别把那三千克装的小馒头往边上一扔，高喊着：这东西根本吃不饱！只要吃自然有吃饱的时候，这没什么难度，只是要意识到这个问题才行；

二、不求记住，一说代码就死记硬背，或者说自己英文不好，学这个肯定不容易……碍的着么？HTML 里常用的单词也就那么十几个吧，还有好几个就是一个字母的，CSS 里常用的兴许有三十个？就再算上 JQ，也就再加上十来个。这么看大概五十单词很难么？而且我们不是闭卷考试的考场，你记不住可以查啊，你要做的就是知道都有些什么功能，我们用手机，其实也是就大概知道有哪些功能，至于在哪里怎么操作其实很多记不清的，用的时候拿着手机找呗，用多了居然就能背下来了，比如当年拿着诺基亚在桌肚里盲打，谁也没觉得这有多难。用的熟了就是自然而然的事情了，那么记得多用用它们哦，至于背，快算了吧，我到现在还是边查边写的，一样写的很开心。

三、我不跟你谈理论，你也别跟我纠结这些理论。去做，先设法达到目的，用的多了就该总结出自己的一套经验来了，然后对照那些理论看看，或者英雄所见略同，或者恍然大悟茅塞顿开，这才是理论的正确用法。否则你开始看这些理论却有完全搞不懂他们，留着做什么？死记硬背应对应试教育么？

四、看懂了，不算会了，因为你未必做的出来；做出来了，不算会了，因为你未必可以解决问题。出问题了，要开心啊，这才是真正学习的机会，遇到问题解决问题才是真正的硬实力。会开车不算什么，堵车了、车坏了、车轮陷住了……你都能应付的了才是真的老司机。

所以多动手做，多去经历问题，是极好的学习方法，遇到问题，你要去思考他的原因，进而猜想他的解决办法，然后验证，如果不行就再猜想，再验证。

自己实在无从下手了，那就搜索吧，你遇到的问题基本前人都遇到过，那么搜索一下，一般都是会有解决方案的。只有当你连搜索的关键词都想不出来的时候才是发问啊，而且问出了结果还要返回来验证和总结。

这里再说一句，遇到别人有问题，而你觉得很简单的时候你回答他对你是有好处的：我许多次以为一个很简单的问题，但是真到了自己去描述他的时候才发现原来还有很多自己意想不到的细节，然后启发自己很多东西，其实回答别人的问题也是一个自我学习的过程。

> 本章学习卡片：卡片 **2**

第三章 几个小故事

在开始之前偶们先来讲几个小故事哦～

假设你一个优秀能干的秘书哦，对，是会干活的那种哦～那天老板给了你一份文档，就是一份普通的 Word 文档， 然后老板希望你把上面的格式规范一下，比方说应该加粗的地方要加粗该设置成标题的地方要设置成标题。那么这种工作其实我们大家都曾经做过的，也没有什么难度，老板一般都会标识出来哪里需要做什么样的修改，我们只要根据他的意思去做就是了。那么其中一段的内容是这个样子的。

> 高端（从这里开始加粗）大气（到这里加粗结束）上档次

那么我们很容易理解出括号里面是说明的内容，说明其他文字应该是什么样的，那么现在的意思其实就是把 “大气” 二字加粗，所以你很轻松的就做出了如下的效果：

> 高端大气上档次

是的，这很简单的，然后第二天老板又给了你一份文档，不过这次他标注的略显简单，毕竟上次标注的内容比实际内容都多，换谁都觉得累啊。

> 低调（开始加粗）奢华（结束加粗）有内涵

然后哩～有了上次的经验这根本不需要猜啊，老板还没转身你就做出来了～

> 低调奢华有内涵

这有什么啊～但是第三天，老板给了你一份董事长写的文档要你整理，这个董事长啊，是个喜欢把中文跟英文掺在一起再加点沙拉酱的讨厌鬼。所以这个文档是这个样子的啊。

> 奔放（start bold）洋气（end bold）有深度

然后你就拿出珍藏已久的神器——中英小词典一查，嘿，合着跟上边一个意思，不过你记住了 bold 是粗体的意思。然后你就做出来啦～

> 奔放洋气有深度

第四天，董事长亲自叼着烟卷给你拿过来一份文档，你一看，嘿，这家伙抽懒筋啊！直接简写了

> 简约（b）时尚（/b）国际范

要是你第一次就看到这样的文档你可能犯懵，可是你经过了上边的千锤百炼之后你就想了啊，看这个情况 b 应该代表的就是 bold，那么其实他就是想给“时尚” 加粗，于是你就明白了 / 代表结束的意思。然后就可以做出

> 简约时尚国际范

董事长满意的夸了你两句，叼着烟卷迈着小碎步离开了。然后我们开始总结一下这几个故事啊。故事都听懂了，但是这么简单的故事啥意思啊？唯一了解到的就是作者很絮叨。

那你看看下面这个啊——

```
狂拽<b>酷炫</b>掉渣天
```

你现在想的一定是—— 你丫再絮叨我抽你！

换个括号就想再说一遍？？？当然了，这次的背景不错……

可是我就再絮叨一句啊～

> 刚才这就是标准的网页代码了啊，他难吗？
>
> 本章学习卡片：卡片 **3**

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com**（刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发： http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书： http://www.jianshu.com/users/d37eaf3de0ff
- 站酷： http://www.zcool.com.cn/u/12927742

第四章 做点准备工作

英文：HyperText Markup Language（我现查的），翻译过来是“超文本标记语言”。

超文本：说明归根结底还是个文本文件。

标记语言：前边都见识到了。

那么记住这两点再学 Html 就简单了，因为你抓住了他的关键。

现在我们要开始做一些准备工作了哦，“工欲善其事，必先利其器”嘛，本来吧，就一个文本文件而已，咱们用记事本也可以无压力的编辑他，但是用记事本阅读黑压压的代码可就真的有压力了，所以我们要用到一款小巧而强大的编辑器 Notepad++（http://notepad-plus-plus.org/zh/）。下载和安装之类的事情我就不去细说了，如果看不懂就一路下一步就好了。

那么，用它有什么好处呢？我们来看一看代码高亮，比如我们看下面一段代码：

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>Document</title>
    </head>
    <body>

    </body>
</html>
```

这样五颜六色的代码就是代码高亮了，他不会影响你的内容，但是用不同的颜色标记之后的内容变的十分易于阅读，我们都是懒人，那么这么舒服的东西自然是要用起的。当然用它的好处可不止这些，我们以后慢慢再讲。

然后很多朋友问我，我用 Dreamweaver 如何？或者我用 Sublime Text 会不会显得更加高大上？我跟你说，都行，都没问题，这就好像我午饭吃馒头还是米饭？都没所谓，因为都能吃饱，至于你说哪个更好吃一些？天啊，我知道你什么口味？吃你自己喜欢吃的，用你自己喜欢用的，总而言之，适合自己的就是最好的。

那么东西准备好了，我来讲两个受益终生的知识点哦：

> - 代码中所有标点均为英文标点，这一点一定要记清，新手最常犯的错误就在这里；
> - 代码缩进（就是上面代码中某些行前边的空白）一定要留好，这个空白是用 Tab 键打出来，空白的长度表示代码的层级，缩进整齐的代码阅读起来会十分轻松；

这两点没看懂也没关系，先有个印象，等到后面用着用着就懂了。

> 本章学习卡片：卡片 **4**

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com**（刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发：http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书：http://www.jianshu.com/users/d37eaf3de0ff
- 站酷：http://www.zcool.com.cn/u/12927742

第五章 一个模板

终于开始要做点什么了，想想还有点小激动呢～

咱们要先写一个基本的模板，或者说一个基础的 Html 文件所必备的一些代码。这就好像呢我们要写一封信，肯定是要先写上称呼的，最后肯定也要有落款和日期，这是一种固定的格式，那么我们现在就把网页文件的固定格式先写出来，至于内容（就好像信的内容）我们以后再说。

打开你的编辑器，比如我们上节课说到的 Notepad++，然后新建一个文件，在其中粘贴上如下代码，然后保存为 index.html

```
<!DOCTYPE html>
<html lang="zh">
    <head>
        <meta charset="UTF-8">
        <title>这里是标题</title>
    </head>
    <body>

    </body>
</html>
```

然后我们来逐行讲解：

```
<!DOCTYPE html>
```

doc 是文档的缩写；

type 是类型；

这里就是声明一下我的文档类型是 html

```
<html lang="zh">
</html>
```

这一对标记是说，我们标记的部分就是 html 代码了。想想我们前边讲的小故事，想想标记是怎么回事哦～

lang 是语言的缩写，zh 是中文，英文的话是 en。就是说页面的主要语言是中文

```
<head>
</head>
```

文件的头部，这里面的信息用来说明这个文档的某些事情。因为头是用来说的啊，

```
<meta charset="UTF-8">
```

第一个要说明的事情来了，

meta 元数据，那么这里说下是什么元数据呢？就是一些说明性的数据，比如文章的元数据，那就是用来说明这篇文章的一些相关信息的，比如作者，比如分类，比如字数，比如创作日期……这里的元数据是用来说明这个网页的一些信息的。

charset 字符集是 UTF-8 ，这个就是编码啊。我们平时说的乱码就是因为编码不对，一个人说英语一个人说中文，不误会才怪呢。

```
<title>这里是标题</title>
```

这个还用说吗？都明摆着写在这里了。

title 是标题的意思，这里是用来显示在浏览器的标题栏（现在都是标签页了）上面的信息。

```
<body>
</body>
```

body 身体啊，这里就是我们写网页内容的地方了，是跟前面 `<head></head>` 相对应的。头是用来说（说明）的，身体是用来干活的。

好像可以看懂了呢~

其实这就是一个模版，初学的时候懂不懂的没关系，每次照抄就行了。

你说什么？背一下？

别开玩笑了，我也没记住，都是复制过来用的……

> 本章学习卡片：卡片 **5**

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com** （刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发： http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书： http://www.jianshu.com/users/d37eaf3de0ff
- 站酷： http://www.zcool.com.cn/u/12927742

第六章 开始写点东西

那么现在，我们理解了标记语言，准备好了一个基础的模版，接下来终于该开始写点什么了，这是一个很美好的时刻，嗯，真的很美好很美好啊～～因为我们终于开始了自己的代码旅程。

其实这个事情真的很简单，甚至比前面的每一章都要简单，为什么呢？因为我们只要在 `<body></body>` 标签之间写下我们要显示的内容就可以了，比如，我们来写代码界最著名的：Hello world！Code is so easy！

好吧，后一句是我自作主张加上的，来看代码～～

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>Document</title>
    </head>
    <body>
        Hello world！Code is so easy！
    </body>
</html>
```

那么把这些代码复制到 **Notepad++** 里，然后保存为 **“**你喜欢的名字**.html”** 之类的废话我就不一次次重复了，还有哦，现在我们都知道了主要的网页代码要写在 `<body></body>` 之间，就像上边演示的一样，那么为了节省篇幅，后面写代码我可能省略掉其他内容，只写 body 标签之间的内容，你们要自己记得把结构补全！

我已经很认真地说过了哦，在后面的章节里要是你再跟我说我就是把你的代码复制进去的（但是你没有补全完整的结构），为啥不起作用？我肯定不会告诉你的哦～（我能说我已经被这么咨询了无数次了么？）

好了，言归正传，简单吧，当然很简单，就是找对位置写一句话而已，于是呢，效果也很简单，就是空白的页面里显示着一句话而已。我们不甘心的哦～想想我们前面学了加粗呢，来试试哦：

```
Hello world！Code is so <b>easy</b>！
```

你说这简直是废话！早就会了啊，你能不能说点新鲜的？好，我们来给它做个分段哦，一句话一个段落。Word 里面我们就是加个回车分隔开两个段落就可以了，那么在代码里这个办法不好用哦，因为回车啊，空格啊什么的是被忽略的（在某些特殊情况下不是）。所以我们要学习一个新的标签 `<p>这里面是一个段落哦</p>`，来看代码：

```
<p>Hello world！</p>
<p>Code is so <b>easy</b>！</p>
```

第一段不用说了，很简单很明白的，来看看第二段，首先是 p 标签标记了整个段落，然后段落里面有 b 标签标记的加粗。这种一层套着一层的情况我们叫做标签的嵌套，记得标签嵌套的原则是：内层标签要完整的包含在外层标签之内，我们来看例子哦：

```
<p>Hello <b>world！</p>
<p>Code</b> is so easy！</p>
```

上面的意思是希望让 `world！ Code` 加粗显示，但是 b 标签却跨越了两个段落，这就错了哦，我们要写成：

```
<p>Hello <b>world！</b></p>
<p><b>Code</b> is so easy！</p>
```

这样才对，那么对此的总结是“你家有老婆，我家有老婆，但是你的老婆是你的老婆，我的老婆是我的老婆，咱俩关系再好，也不可以共用一个老婆！”那么一说一笑就记住了哦～～

> 本章学习卡片：**卡片 6**

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com**（刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发： http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书： http://www.jianshu.com/users/d37eaf3de0ff
- 站酷： http://www.zcool.com.cn/u/12927742

第七章 认识链接

我们学会两个标签了，`<b></b>` 可以给文字加粗，`<p></p>` 可以标记段落，但是只有这些的网页跟 Word 有什么区别？所以我们要研究一下互联网的一个重要组成部分——超链接。

现在还有几个人会去说超链接这个词呢？还有久违了的万维网，想到这些不禁唏嘘时光荏苒。当初夸耀的荣光无限的互联网，现在也被我们习以为常了。那么被我们视为高深莫测的代码，在明天会不会也变成家常便饭……当然不会了，因为技术总是要越来越易用的，但是编程却在不知不觉中容入了我们的生活，比如可以预约的电饭煲，其实对他的设定也可以看作简单的编程……

一不小心跑题了，回到已经不再 Super 的链接上来。老板说：你把这段话里公司的名称都链接到咱们的网站去，让别的公司也看看，咱们也是有网站的企业了，不许他们再说咱们土……了。这时候你就想啊，稻米鼠的教程写的太慢了，还没讲……

其实链接的标签很简单的哦，就是字母表第一个字母 a ，你说你会写了，那肯定就是 `<a></a>` 了。那么仔细想想啊，刚才老板叫你干啥了？把公司的名字链接到咱们的网站啊～咱的网址你写上了么？

我来举一个最简单的链接的例子啊：

```
<a href="http://coffee.zji.me/">代码能有多难？</a>
```

你说嘿，还真看懂了，href= 后面写上网址就对了。嗯，`<a href="要链接到的网址">要显示的文字</a>` 这就懂得了。那么我们来讲一讲标签的属性。

标签是有属性的，就好象说这个英雄攻多少，防多少（玩游戏的男生能理解哈），就好象说这个化妆品是保湿的还是防晒的，防晒值多少（爱美的女生能理解哈），这些都叫做属性，那么我们标签的属性也是一回事，就是对标签一些相关的内容进行说明和定义，你说你这是链接，那么链接到哪里就是一个属性，来看看下面的格式哦：

> <标签 属性="属性值"></标签>

来对照上面的代码基本就记住这个格式了哦，你没记住？也没关系的，反正往下看慢慢就记住了。

老板检查了一下你的成果，感觉虽然链接成功了，但是满足不了他那土的掉渣的审美，说：那个链接要在新标签里打开～

这时候我们怎么办？加个属性，定义一下在哪里打开就是了。于是我们把代码改成了：

```
<a href="http://coffee.zji.me/" target="_blank">代码能有多难？</a>
```

target 这个属性说的是链接的打开位置，默认是在当前页面打开的，如果要在新页面打开，属性值为 _blank ，就像上边一样，target 还可以有一些其他的值（链接还有一些其他的打开位置），但是最常用的就是 _blank（新标签打开），记不住？别记了，用的时候到这篇文章查询就是了，反正这么常用的东西，查个十几次之后你总会记住的，是不是？

你说链接还有什么好玩的属性？有啊，title 属性可以给链接加上说明文字的哦，比如了：

```
<a href="http://coffee.zji.me/" target="_blank" title="这是一本写给小白们的看得懂的，学得会的代码书">代码能有多难？</a>
```

你说试过之后发现没啥变化？那你把鼠标放在链接上不动，待一会就会显出说明文字来了，鼠标移开说明消失。啊，这种体验大家都很熟悉了是不是？原来就是这么写出来的，原来就是这么简单！

> 本章学习卡片：卡片 **7**

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com** （刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发： http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书： http://www.jianshu.com/users/d37eaf3de0ff
- 站酷： http://www.zcool.com.cn/u/12927742

第八章 半个标签

现在我们学过的标签都是一个开始，一个结束，为什么腻？因为其实我们要标记的是一个范围，所以要给出一个开始和一个结束的点。那么有没有我们只需要给出一个点的情况呢？当然有了，来看两个标签先：

> `<br />`
>
> `<hr />`

解释一下哦，`<br />` 是在这里换行，那么明显我们只需要指定一个位置就可以了；`<hr />` 是在这里插入一条分割线，当然也只要告诉他一个位置就够了。

既然只需要一个位置，那么写两个（一组、一对）标签就是浪费时间，所以懒惰的程序员们就把这一对标签合二为一，看看上边的写法，明白了吧。等等，不是什么标签都可以这么合并的哦，我这么说只是为了让你理解一下的，你可别光顾着撸串又把我的话当了耳旁风啊～

这个当然没什么难的，不过有个很著名的标签也是独行侠哦～～他独来独往，至今还是一只单身狗，那么他就是大名鼎鼎的图片标签 `<img src="在这里写图片地址哦～" />`

那么现在，我们来玩哦～我这有一个图片地址： `http://coffee.zji.me/imgs/bridge.jpg` ，那么我们的代码应该是：

```
<img src="http://coffee.zji.me/imgs/bridge.jpg" />
```

效果如下哦：



那么看懂啦 src 属性就是要插入的图片的地址。然后我们再看看他的其他属性哦，这个开始有意思了。

```
<img src="http://coffee.zji.me/imgs/bridge.jpg" alt="这里是图片的说明文字" />
```

alt 很像链接里的 title 属性，这个是用来说明图片的，但是他还有一个很重要的作用，要是因为网络等问题图片没能正常显示的时候，这些文字会显示在图片位置，让人们知道这里应该是什么。效果如下：



上面的这些对照着链接都很容易理解了，但是图片还有两个常用的属性宽（width）和高（height），一般的说，我们给他们指定的值就好了，比如呢～

```
<img src="http://coffee.zji.me/imgs/bridge.jpg" width="300" height="300" />
```

在 html 代码里，如果没写单位则默认单位是 px（像素），效果如下：



而如果你只写了其中一个属性，则图片会按比例缩放，也就是另一个属性会跟着等比例变化。比如：

```
<img src="http://coffee.zji.me/imgs/bridge.jpg" width="300" />
```

它的效果如下：



那么我们基本了解了图片这个标签，现在我们来为图片加个链接吧~

```
<a href="http://coffee.zji.me/">
    <img src="http://coffee.zji.me/imgs/bridge.jpg" />
</a>
```

好啦，那么这次就聊到这里，记得给我买咖啡喝哦~

> 本章学习卡片：卡片 **8**

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com**（刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发：http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书：http://www.jianshu.com/users/d37eaf3de0ff
- 站酷：http://www.zcool.com.cn/u/12927742

第九章 来吧，表哥（表格）！

> 如果本章演示效果无法正常展示，请至主站查看 http://coffee.zji.me/

可惜我不是表妹……

不搞笑了，表格这一章其实很简单，只是很热闹，不过好在现在我也不怎么用表格，大家看懂就好，当然要是做淘宝的朋友们，我还是建议你认真学习一下，毕竟在淘宝上表格用的还是很多的（用作布局）。

是的，用表格可以布局的，或者说排版，随你怎么叫，反正都是一回事，不过现在几乎被淘汰了，大家都用 DIV + CSS 布局去了，有人说你这个叫法不科学！反正科学不科学的我不是很清楚，我就知道现在招聘前端都煞有介事的写上：熟练掌握 DIV + CSS 布局；熟练掌握 Dreamweaver …… 说得好像我们这些不用 Dreamweaver 的多么山寨，多么不专业似的。

不吐槽那些应该吐槽的了，然后强调一下：写代码一定要把代码缩进写整齐！！！重要的事情三个感叹号哦～然后让大家理解一下什么是清晰的代码缩进，下面代码中我把每一个 tab 缩进换成一个波折号：

```
<!DOCTYPE html>
<html lang="zh">
— <head>
— — <meta charset="UTF-8">
— — <title>这里是标题</title>
— </head>
— <body>
— — <a href="http://coffee.zji.me/">
— — — <img src="http://coffee.zji.me/imgs/bridge.jpg" />
— — </a>
— </body>
</html>
```

看，这样各个标签间的包含关系和层次是不是一目了然了？这个一定要理解，也要做到；这个习惯是对于任何阅读你代码的人的起码的尊重，乱成一团麻的代码没人爱看，而且你自己阅读起来也很吃力。最后用当初我的老师的一句话来结束这个话题：

> 代码没有清晰缩进的作业别给我看，我看不懂！

好了，开始我们今天的话题，表格，刚才为什么要强调缩进呢？因为表格的标签层次比较多，要是不清晰缩进的话，一个表格没写完你就哭昏在加班的午夜了。

首先，表格的标签是 `<table></table>` ，但是这还不够，我们还要知道行的标签是 `<tr></tr>` ，和单元格的标签是 `<td></td>` 。然后怎么把他们合理的放在一起呢？——按着大小个嵌套：

```
<table border="1">
    <tr>
        <td>第一行第一列</td>
        <td>第一行第二列</td>
        <td>第一行第三列</td>
    </tr>
    <tr>
        <td>第二行第一列</td>
        <td>第二行第二列</td>
        <td>第二行第三列</td>
    </tr>
</table>
```

效果如下：

第一行第一列	第一行第二列	第一行第三列
第二行第一列	第二行第二列	第二行第三列

需要解释的有两点：第一、border 这个属性是指边框的粗细，我设置了 1 ，这样就能看到边框了，因为默认是 0 ，那样就不容易看懂效果了。第二、每一行中的单元格数量是相同的，合并单元格的效果在下面讲；

单元格有两个跨越属性：colspan（跨列）、rowspan（跨行），是指当前单元格占用几列，或者占用几行，我们来举例：

```
<table border="1">
    <tr>
        <td colspan="2">第一行第一列</td>
        <td>第一行第二列</td>
    </tr>
    <tr>
        <td>第二行第一列</td>
        <td>第二行第二列</td>
        <td>第二行第三列</td>
    </tr>
</table>
```

这个单元格要占两列的位置，效果如下：

<table>
<tr><td colspan="2">第一行第一列</td><td>第一行第二列</td></tr>
<tr><td>第二行第一列</td><td>第二行第二列</td><td>第二行第三列</td></tr>
</table>

然后再看跨行：

```
<table border="1">
    <tr>
        <td rowspan="2">第一行第一列</td>
        <td>第一行第二列</td>
        <td>第一行第三列</td>
    </tr>
    <tr>
        <td>第二行第一列</td>
        <td>第二行第二列</td>
    </tr>
</table>
```

第一个单元格要占用两行，所以效果如下：

<table>
<tr><td rowspan="2">第一行第一列</td><td>第一行第二列</td><td>第一行第三列</td></tr>
<tr><td>第二行第一列</td><td>第二行第二列</td></tr>
</table>

好复杂是不是？我也这么觉得，所以一般我都是用可视化编辑器（比如 Dreamweaver）来生成表格，极少自己手写，上边几个属性也是现查了抄过来的。大家看看做个了解，将来看到能看懂就很好了。

当然很多做淘宝的朋友会问了，虽然说看懂就行，但是我还常见到一个标签你没说啊——`<tbody></tbody>`

这里也说一下仅作了解：其实有三个标签 `<thead></thead>` 这是表头，诶，这你听懂了，那么下面就好理解了，`<tbody></tbody>` 是表格的主体，那么 `<tfoot></tfoot>` 就是表格的注脚（我也不知道怎么取名字，知道我什么意思就好）他们的嵌套顺序是酱紫的：

```
<table>
    <thead>
        <tr>
            <td></td>
            <td></td>
        </tr>
    </thead>
    <tbody>
        <tr>
            <td></td>
            <td></td>
        </tr>
    </tbody>
    <tfoot>
        <tr>
            <td></td>
            <td></td>
        </tr>
    </tfoot>
</table>
```

有的时候我在想，我为什么要讲表格？后来想想，淘宝设计师还是用的到的，那么只好咬牙坚持了。讲表格要做噩梦的，求安慰啊～～

> 本章学习卡片：卡片 **9**

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com**（刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发：http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书：http://www.jianshu.com/users/d37eaf3de0ff
- 站酷：http://www.zcool.com.cn/u/12927742

第十章 表格布局原理

> 如果本章演示效果无法正常展示，请至主站查看 http://coffee.zji.me/

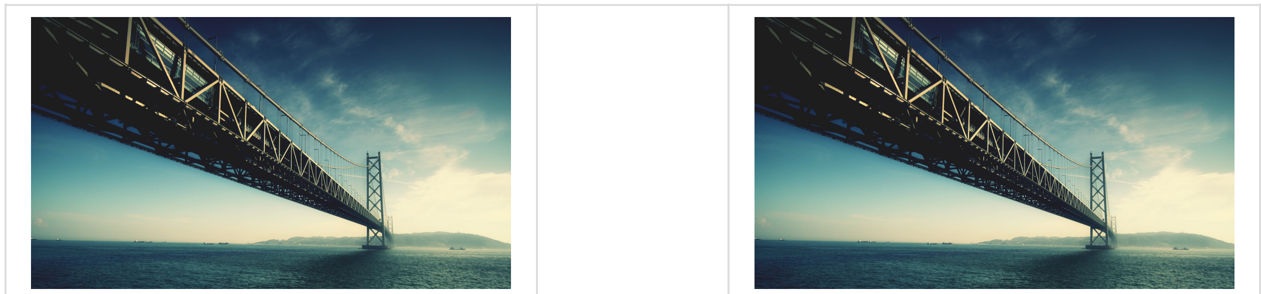
这个吧，其实真的没有很复杂的道理，就是每个单元里放上相应的内容，靠表格控制着他规规矩矩的呆着。当然了，没东西的地方就不填写了，于是就留出了空白。

现在很少有人用表格去布局了，除了淘宝之类的特殊情况，或者……这家伙不会 CSS。因为表格不灵活，修改起来也十分的麻烦啊～在这里简单讲讲做个了解。

比如我现在有两张图片，想让他们排在一行，然后相互距离 100 像素怎么办呢？我们就做一个一行三列的表格，第一个单元格和第三个单元格分别放一张图片，而第二个单元格留空，设置宽度为 100px。代码如下：

```
<table>
    <tr>
        <td><img src="http://coffee.zji.me/imgs/bridge.jpg" width="300" /></td>
        <td width="100"></td>
        <td><img src="http://coffee.zji.me/imgs/bridge.jpg" width="300" /></td>
    </tr>
</table>
```

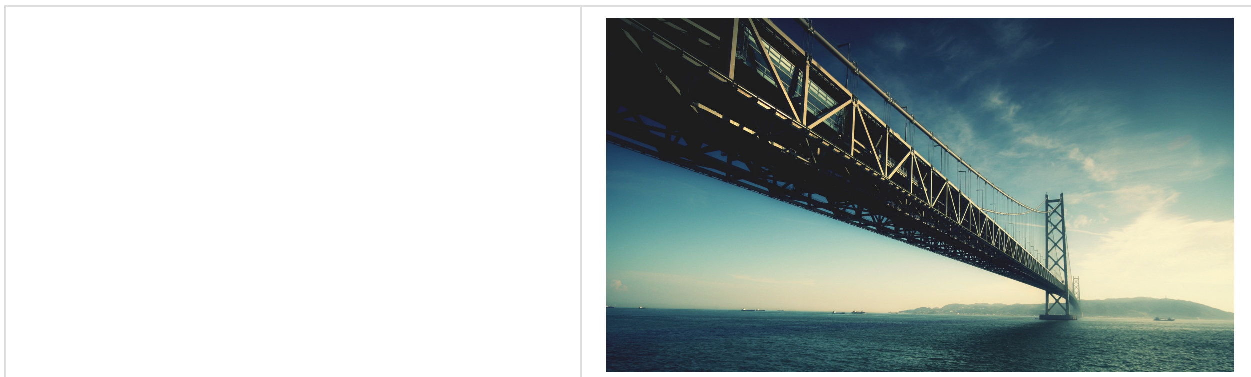
来看看效果：



那么我们只是想让一张图片距离左侧 300px 呢？

```
<table>
    <tr>
        <td width="300"></td>
        <td><img src="http://coffee.zji.me/imgs/bridge.jpg" width="300" /></td>
    </tr>
</table>
```

效果如下：



所以这个东西其实很好理解，设置合适的单元格尺寸，来撑开布局而已，你用 Excel 也可以玩。然后到了布局复杂的不行了，大家又发明了表格的嵌套，就是把一个表格放到另一个表格的单元格里，乱的让人想吐......不多说，只做了解。下一章我们学习一个重点的内容哦——表单。

> 本章学习卡片：卡片 **10**

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com**（刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发： http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书： http://www.jianshu.com/users/d37eaf3de0ff
- 站酷： http://www.zcool.com.cn/u/12927742

第十一章 一起写个百度

> 如果本章演示效果无法正常展示，请至主站查看 http://coffee.zji.me/

今天我们来聊聊表单，这个名字可能有些朋友不太了解，啥是表单呢？就是在网页上需要你填上东西然后点个按钮把你填写的东西提交了的功能就是表单了。这么说好抽象（可怜我还没去抄定义，努力通俗的去说呢），但是我举个例子你就该明白了：搜索框啊，网站注册啊，网站登录啊，都是由表单来实现的。还不明白？那我们一起来写一个小小的示例哦～

首先写下：

```
<form action="http://www.baidu.com/s" method="get">
    <input type="text" name="wd">
    <input type="submit" value="搜索" />
</form>
```

然后看一下效果：

搜索

曾经以为很高深复杂的东西原来也不过这么几行代码，现在我们来仔细分析一下这些代码哦。

form 这个标签包含的部分叫做表单域，反正就是这块地盘是它的，它做主，它罩着，有事冲他说。

然后 action 这个属性是一个网址，是用来处理数据的网址，这个太高深，我们先不要理解它，反正就是我们把收集到的这些数据发给谁的意思。你在大街上做社会调查，别人给你把表填好了，你得把表拿回去给统计部门的人，让他们研究去，嗯，这里就是说这个表要交给谁。

method 是提交的方式，分为两种，一种是 get，一种是 post。

get 就是把数据放在网址里传递，下面我们会做演示。那么打个比方啊，你在外面做了调查，回来之后直接高喊一声：统计部的王二（这是是要让谁处理，就像 action 的网址）我把调查的情况跟你口头说一下（跟着前边招呼负责人的话就一起说了）。这种方法方便（张口就说啊），私密性差（边上的同事都听得到），可以传递的数据量小（三十万字的内容你口头报告一个试试）；

post 就是用一些我现在跟你说不明白的办法传递数据。但是打个比方你就能理解了：post 是啥意思？初中英语老师告诉我说是邮寄的意思，那我们就这么理解，你做完了调查，直接把调查结果寄给负责人。会有什么特点呢？肯定要比口头汇报复杂一些，但是安全性高（相对的啊，快递会不会在中间在做手脚是个问题，但是目前我们先不考虑这个，起码同事没法伸

个耳朵就偷听走了），可以传递的数据量大（调查的时候我看见一女孩，长得那叫一个正点，偷拍一张跟朋友分享，夹在数据报告里一起邮寄就可以了，你口头汇报描述一个试试看，跟图片完全不是一个感受了）；

input 是表单项，他又很多类型（type），设置不同的类型（type）可以实现不同的功能。就好象都是象棋，车和炮就有不一样的功能。

text 这个类型就是文本框，就是上面效果中那个可以输入文字的地方，那叫作文本框，多讲理，可不就是用来输入文本的方框嘛～

submit 这个类型是提交，就是把收集到的信息交给处理人。换言之老板喊收工啦，把调查结果都交上来就可以下班啦，就这个意思。

name 是这个表单项收集到的数据的名称。就是标记一下，这一栏写的是姓名，那一栏写的是年龄，嗯，就这个意思，但是要用英文。

value 就是这个表单项的值，在这里我们用他设置了提交按钮上的文字，你可以修改它看看变化。

这些属性讲完了，我们来试用这个小小的搜索工具哦，在里面输入 AAA 很好识别的一个字符串（就是一串字符，叫做字符串）。然后提交，看看页面跳转到哪里了？

> http://www.baidu.com/s?wd=AAA

然后我们仔细看看这个网址，是不是觉得似乎看明白了呢？`http://www.baidu.com/s` 是 action 的值，然后一个问号，然后是文本框的 name 值（wd）等于文本框的值（我们输入的 AAA）。

原来写个简单的搜索就这么几行代码的事情，是不是有点想要用它做些什么了呢？

> 本章学习卡片：卡片 **11**

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com**（刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发：http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书：http://www.jianshu.com/users/d37eaf3de0ff
- 站酷：http://www.zcool.com.cn/u/12927742

第十二章 相对地址

好了，现在我们算是初步的认识了 Html 代码，好像是叨叨了好多章，其实就是一个中心——标记语言。记住了这一点 Html 还是十分容易理解的，也因为它很容易，我就不在这里过多的废话了（虽然已经被无数人指责废话成灾了）。从下一章开始我们要学习神一样的 CSS 了。

好了，先来学点基础知识准备后面充满高能的战斗，我们来研究一下文件地址。首先记住这章很有用，但是现在先看懂就好。

文件的地址分为两种：绝对地址和相对地址，现在我们来打个比方，我最喜欢这个时间了。

北京东城区五星级别墅区里面 A 座是你家，家里三个房间，房间一里面有一张桌子，桌子上有个花瓶。

你站在房间一里，我问你，花瓶在哪？你回答我："北京东城区五星级别墅区 A 座房间一桌子上的那个花瓶"。这是绝对地址，这个地址你在哪里告诉对方，对方都可以凭这个地址找到这个花瓶，很完美，就是太罗嗦。

所以一般我们都会简单地说："我身边桌子上的这个花瓶"。一样很明确，但是有个前提，我得知道你在哪里，才能找到这个花瓶。因为这是以你为参照物的，找不到你，那么花瓶的位置也无从谈起。然后你不能换位置，如果你跑到我家说这句话，那指的就是我家的花瓶了。所以要想让这个相对地址成立，你和花瓶的相对位置不能变化。

你说这么看来相对地址除了简单也没啥好处啊，不，当你整体移动一些相关联的文件的时候他的作用就显现出来了。比如还是刚才那个状态，我把你家别墅（包括桌子花瓶还有房间里的你）整体打个包交给顺丰寄到上海去了。到了上海打开一看，房间一里面一张桌子，桌子上是花瓶，你站在桌子旁边，一切都没变（也就是相对位置没变哦），那么你现在说："我身边桌子上的这个花瓶"，就还是这只花瓶没错，但是你再说："北京东城区五星级别墅区 A 座房间一桌子上的那个花瓶"，那就不好意思，在那个位置我找不到花瓶了，因为连房子我都给你搬到上海了。

现在理解了绝对地址和相对地址的一些特点，你就开始有些明白了，那么我们说：**"绝对地址是描述某个文件的位置，而相对地址是描述两个文件间的位置关系"**。当我们移动一个文件夹的时候，里面的每个文件的地址都发生了变化，但是他们之间的相互位置关系却没有变。

然后我们来做一个实例来讲解一下哦，假设有如下目录结构：

`http://coffee.zji.me/`

- |_ imgs
    - |_ shuaige.png
- |_ files
    - |_ miao

        - |_ huamao.jpg
    - |_ index.html
    - |_ photo.jpg

那么简单解释一下：

> `http://coffee.zji.me/` 这个网站下有两个文件夹 `imgs` 和 `files` ；
>
> 其中 `imgs` 文件夹下只有一张 `shuaige.png` 的图片；
>
> `files` 文件夹下有两个文件 `index.html` 和 `photo.jpg` ，还有一个叫做 `miao` 的文件夹；
>
> `miao` 文件夹下有一张图片 `huamao.jpg`

好了，我们假设你读得很认真，你完全理解了我上边所说的这些乱七八糟的东西，那么下边便是很轻松的学习关键技巧的时刻了。如果你看到这里发现上面的都没看……那就记住下面的方法，用的多了自然也就明白了。

现在我们都以 `index.html` 为参照物（其实就是在哪个文件里写就以哪个文件为参照物），我们在 `index.html` 里面写 `photo.jpg` 的相对地址。看我怎么做：

先写下两个文件的绝对地址：

> http://coffee.zji.me/files/index.html
>
> http://coffee.zji.me/files/photo.jpg

然后去掉两者前边的相同部分，得到：

> index.html
>
> photo.jpg

现在我们的参照物（ **`index.html`** ）只剩下文件名了，那么下面的就是我们要的相对地址了—— `photo.jpg`

然后做个练习，写 `huamao,jpg` 在 `index.html` 中的相对网址，一样的办法，先写下他们的绝对网址：

> http://coffee.zji.me/files/index.html
>
> http://coffee.zji.me/files/miao/huamao.jpg

然后去掉相同部分：

> index.html
>
> miao/huamao.jpg

`index.html` 的网址就剩下文件名了，那下面的就是我们要的相对地址了——`miao/huamao.jpg`

这个简单到混蛋的办法是不是很好理解？我们来看点复杂 `shuaige.png` 在 `index.html` 里面的相对地址怎么写？

按着上边的办法，列出绝对地址：

> http://coffee.zji.me/files/index.html
>
> http://coffee.zji.me/imgs/shuaige.png

去掉相同部分：

> files/index.html
>
> imgs/shuaige.png

嘿，`index.html` 现在剩的可不只是文件名，还带着一个文件夹呢，去重之后，如果参照文件剩下的不只是文件名，则在参照文件（**`index.html`**）上去掉一个文件夹，同时给目标文件（**`shuaige.png`**）前面加上一个 `../` 。如此重复操作，直到参照文件（**`index.html`**）只剩下文件名为止。

说起来很复杂，做起来就简单了，简直是无脑的上面删一个，下面加一个：

> index.html
>
> ../imgs/shuaige.png

这样就出来了。现在解释一下 `../` 代表向上一层文件夹，那么如果是 `../../` 就是向上两层喽～现在看看这两种方法，想想上面的比喻，好好消化一下，可以不理解，但是要知道我都说过些什么，等到后面用起来就理解了。

最后补充一句：你说网址的我看明白了，但是在本地的时候怎么写？一样的 **`C:\windows\我是大帅哥.avi`** 这么写文件绝对地址，其实你在文件上面右键属性就可以找他的绝对地址了。然后按着上边的方法做哦，唯一的问题是 `\` 要换成 `/` 。

> 本章学习卡片：**卡片 12**

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com** （刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发：http://coffee.zji.me/

- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书： http://www.jianshu.com/users/d37eaf3de0ff
- 站酷： http://www.zcool.com.cn/u/12927742

第十三章 神一样的 CSS

> 如果本章演示效果无法正常展示，请至主站查看 http://coffee.zji.me/

当然这只是一个比喻，我绝没有挑起代码界圣战的意思。从 Html 进入到 CSS 世界的感觉就好像从人间一步迈进了神国，一些曾经很难做到的事情，现在变得信手拈来了。宛如我们掌握了神的力量。当然这么说下去的话后面我们要学的 JavaScript 就应该称作众神之神了……

玩笑开完了，玩代码的时间到了，我们先来简单认识一下 CSS。

首先看一下我们前边写的代码：

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>Document</title>
    </head>
    <body>
        <p>Hello world！</p>
        <p>Code is so easy！</p>
    </body>
</html>
```

挺简单的代码，效果如下：

Hello world！

Code is so easy！

就是两段文字，现在我给他添加一点属性哦：

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>Document</title>
    </head>
    <body>
        <p style="color:red;font-size:12px;">Hello world！</p>
        <p>Code is so easy！</p>
    </body>
</html>
```

大家看到，我就是给第一个段落加了一个属性。来看看效果：

Hello world！

Code is so easy！

我们来分析代码哦，`style` 是样式的意思，那么很容易明白，这个属性就是给当前的元素加上一些样式。那么我们加了什么样的样式呢？`color:red;` 颜色是红色，颜色（color）是一个 CSS 属性，而红色（red）是这个属性的值。哎，看就跟 Html 有区别了，Html 是 `属性=属性值` ，而 CSS 是 `属性:属性值` 。（特别强调：都是英文标点哦）。然后 CSS 的每个属性结束都要用英文的分号标记出来，所以 `font-size` 就是第二个属性了，意思是字体的尺寸，属性值是 `12px` ，于是我们就看到了上边的效果。

所以看到了，CSS 也不难，不过是换个格式去写属性，属性值。说来说去也就像代数里的赋值操作，然后就没有然后了。

那你说这个 CSS 有啥好的啊？方便！来看哦，现在代码里有两个段落，那么我们假设我们有一百个段落（我不写了，就拿这两个代表了）要是我想都设置成刚才那样，咋弄？把 style 属性整个复制，然后挨个 `<p></p>` 标签里粘贴？好不容易粘完了，老板告诉你改回第一稿......

来看我怎么写：

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>Document</title>
        <style>
            p {
                color:red;
                font-size:12px;
            }
        </style>
    </head>
    <body>
        <p>Hello world！</p>
        <p>Code is so easy！</p>
    </body>
</html>
```

效果如下：

Hello world！

Code is so easy！

好了，再来分析代码哦～

我们说了 `<head></head>` 标签是用来声明东西的，那么我在他里面声明一下这个页面的样式他应该不会有什么意见，就是本职工作啊。但是我们要把这些样式放在一起，不许他们乱跑，所以写上 `<style></style>` 标记一下，这里面的都是样式，就是都是 CSS 代码啊，你要

按着 CSS 那么处理。于是这个事情明白了。

再看里边，`p` 代表标签 `<p></p>` 这个猜都能猜出来，是吧，后面是他的样式（color，font-size），我们前边也都认识过了。现在的问题是大括号，大括号就是说这部分代码都是前边 p 的啊，又是一前一后标记一下，这么简单的事情我们一想就明白了。

看到啦，我在 CSS 里说 `p {……}`，于是页面里所有的 `<p></p>` 标签就都获得了这个样式，很方便是不是？

——“老板，那一百段的文字颜色我都改好了！”

——“这么快！没骗我吧，正好我这还有点活，你就能者多劳吧……”

CSS 这么厉害的东西确实是能者多劳，现在我们一般只用 Html 写网页的框架和内容，而样式全部交给 CSS 处理，他清晰明了，便于管理和修改，而且也十分得强大。

但是能者多劳就该出问题了，你想啊，CSS 管理了整个页面的样式，有着很多内容，全放在页头，我们想看网页的 Html 结构的时候就要往下找很久。这就好像把档案管理员跟老板放在一个办公室，老板迟早要被堆积成山的档案埋没（为什么说到这里我很开心呢？）。

然后还有一个问题，你想我写一个网站，肯定很多样式是相似或者相同的，那么这部分咋整？每个页面复制一份？然后老板说了，吧所有页面的文字都换成蓝色，你又挨个页面复制粘贴……

所以我们要把 CSS 独立出来，这样他不会影响我方便的阅读 Html，还可以在多个页面里调用，谁用他就叫他，我们共用一个就好啦。是啊，公司就一个档案管理员，不是一个部门塞一个，那样机构臃肿，职能重叠，效率低下……

所以我们新建一个 css 为后缀的文件，比如叫做 `style.css`，文件名你随意啊，然后把这部分代码粘进去保存。

```
p {
    color:red;
    font-size:12px;
}
```

这就是独立的 CSS 文件了。然后我们把我们的 Html 代码改成：

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>Document</title>
        <link href="mycss/style.css" rel="stylesheet" type="text/css">
    </head>
    <body>
        <p>Hello world！</p>
        <p>Code is so easy！</p>
    </body>
</html>
```

我们发现这就是在普通的 Html 文档上加了一个 link 标签，我们来分析一下，link 是链接的意思，就是我们要把我们的 CSS 文件链接到这个文档来，要不然浏览器怎么知道你要用哪个 CSS 文件呀。所以这是一个声明，放在 head 势力范围内没错。

现在研究 link 的属性哦，href 是链接到的地址，就是我们要用的 CSS 文件地址，我这里写的是相对地址，所以你不要光是照着抄，要写你自己那个 CSS 文件的相对地址哦，否则找不到文件自然就出错了。

rel 属性是说被链接文档和当前文档之间的关系，就是样式表（stylesheet）喽，顺便说一下 CSS （Cascading Style Sheets）的中文名称是“层叠样式表”，简称“样式表”。

然后 type 是类型，text/css 说的是这是一个文本文件（text），同时也是一个 css 格式的文件。

当然这种格式也不用记，跟我一样用的时候抄一下就好了。所以呢，今天我们学习了 CSS 在 Html 文件中的三种出场方式和一些 CSS 的优点。大家可以看学习卡片中的总结再复习一下主要知识点哦～

> 本章学习卡片：卡片 **13**

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com** （刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发： http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书： http://www.jianshu.com/users/d37eaf3de0ff
- 站酷： http://www.zcool.com.cn/u/12927742

第十四章 从 div 扯开去

> 如果本章演示效果无法正常展示，请至主站查看 http://coffee.zji.me/

我们前边说过 div+css 布局是招聘时一种懒到家的衡量标准（其实现在 css 已经是前端的基本技能了），就好像招聘文员写上要求会打字一样的多余。好吧，虽然这个说法被无数人吐槽着，也有些人较真说这个说法不科学等等。但是我们学习 css 总是绕不开要用到 div 的。

其实 div 就是一个普通的 Html 标签，普通到什么地步呢？他几乎没有任何自带的样式，好吧，他就是一个空白的标签（是样式是空白的）。一张白纸啊，做什么都好用。我们知道 CSS 的作用就是定义样式啊，如果我们使用的标签本身自带了样式，我们还要用 CSS 对他进行修改，使其达到我们想要的效果。但是有了 div 就方便了啊，我们只要把我们需要的样式给他就好了。毕竟新建总是要比修改来的清爽省心。就好像拿张白纸写字很舒心，但是拿张写过字的纸过来，你要先把文字擦掉再去写你想要的文字，就比较闹心了。

然后说说盒模型，这东西搞了个高大上的名字唬人而已，其实就是把 div 看成一个纸箱，那么一个页面很多 div 就是一堆纸箱在房子里排开而已。进而的，div 的嵌套就是大箱子套小箱子。就这么回事，比如下图就是本教程主站的盒模型 3D 视图：



就是像摆积木一样摆出了一个页面，你说要是真的用积木去摆就好了，你就会了。不过其实换做代码操作也不过是你用电脑能听懂的语言把各个积木块的样子和位置描述一下罢了。

好像......有点乱哦，你想 div 这个空白标签看起来挺好用的，可是要是页面上全用他，那可咋区分啊？我在 CSS 里面写上 `div {......}` 结果所有 div 全都一起发生变化了，这就有点乱了。所以我们要给 div 取名字。

其实呢，所有的 Html 元素都可以有名字的，这个名字分为两种 `class` 和 `id` ，用法很简单，就是当做标签的一个属性去写：

```
<div id="myid"></div>
<div class="myclass"></div>
```

就是这样，当然他们的名字你随便取，不过别搞怪，写中文啊，特殊符号啊什么的肯定不行。那么你又问了，取名字就取名字吧，怎么还两种呢？这个也好理解， id 是学号，class 是班级，反正我们初中的英语老师（为什么不是小学？我们小学没有英语课呀）就是这么告诉我们的。

然后我们该说说 id 和 class 的区别了，他们都可以用来标识某个元素，但是他们是有区别的，否则我们也不用搞两个名头出来。你知道的，在学校里一个学号对应一个学生，你报出学号就能找出有且只有一个学生。所以老师提问的时候爱叫学号，肯定有人被叫到（他叫范围以外的我们就不鸟他了）而且被叫到的肯定是一个人，不会半个班都站起来跟他吼："报告老师，我不会！"而班级呢？一个班级可以有很多学生，这些学生的特点就是，他们是一个班的（废话！），一个班的应该就是一个年级的，应该就是一样的年龄，一样的专业……反正是一群在某方面属性一致的家伙。

所以， 在同一页面内， id 是不可以重复的，不能有两个元素具有相同的 id。某学校的两个学生学号一样，麻烦了，520 号考上了哈佛，结果两个人，谁去谁不去？就打起来了。

但是，在同一页面内，可以有多个元素具有相同的 class，三年二班有三十个同学，这很正常，校长说，让三年二班的同学去表演合唱，那么这三十个同学就都在台上唱歌，整齐划一，很好看。

这就是两者的特点。然后说，我们到 CSS 里面怎么写呢？这就涉及到了选择器的问题。先说啥是选择器，我说页面中某个元素怎样怎样，那么究竟是哪个元素呢？选择器就是用来告诉我们到底是哪个元素的。比如：

```
p {
    color:red;
    font-size:12px;
}
```

这是我们上节课的代码，大括号里是一些 CSS 属性，我们不去管他，前边的 p 是说这些属性对所有的 `<p></p>` 标签起作用，那么这个 p 就是一个选择器。

然后要是想对链接起作用呢？就是 `a {……}` ，于是你看出门道来了，原来选择器就是元素的标签啊，那么 `div {……}` 这么写的话，大括号里的 CSS 属性肯定就对所有的 `<div></div>` 起作用啦？

没错，就是酱紫的！你真聪明～

但是你肯定不会就这么依了我的，刚才我们给 div 取了半天名字，然后到这根本没用到啊。放心，我们接着就来学习 id 和 class 在 CSS 里面怎么去用。

其实也很简单的，还是上边那段代码，我们给里面加一点点文字用来看效果哦：

```
<div id="myid">我是文字，用来看效果的</div>
<div class="myclass">我也是</div>
```

然后 css 写作：

```
#myid {
    color:blue;
}
.myclass {
    color:red;
}
```

效果如下：

我是文字，用来看效果的
我也是
这你就明白了，在名字前面加个 `#` 号就代表 id，在名字前面加个 `.` 就代表 class。用 `#` 来开头作为编号什么的我们很常见，这就记住啦。class 是个很平常的东西，不像 id 那么独一无二，所以这么平凡大众的属性就没得炫耀了，点个点低调的路过就好。

今天我们认识了 div，还知道了班级（class），学号（id）的特点与写法。明天我们就开始试着去写一些东西喽～

> 本章学习卡片：卡片 **14**

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com** （刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发： http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书： http://www.jianshu.com/users/d37eaf3de0ff
- 站酷： http://www.zcool.com.cn/u/12927742

第十五章 开始征途

> 如果本章演示效果无法正常展示，请至主站查看 http://coffee.zji.me/

我说过学代码要勤动手，其实你们光拿我当相声看了，根本没动手。别以为我不知道哦。今天开始我们要一起写一个页面了，大家可一定一定要动手跟着操作才行啊，否则你会跟不上的！

> 看懂和会做绝对是两码事！

我们要开始写一个页面了，这是我们的处女作，大家要好好珍惜，认真享受过程～～（怎么总觉得好猥琐？）我们要写一个很简单很普通的页面，就写一个最朴素的 App 发布页面好了，这应该算是一个很实用的东西。我就不画效果图给大家看了，因为这又不是给老板干活，花毛线效果啊，做成什么样是什么样就好了嘛～

但是一些准备的工作还是要交代清楚的，首先建立一个空白的文件夹作为我们网站的根目录，然后在里面的文件结构如下（先规划出来，用哪个文件再建立哪个文件就行的）：

/

- |_ images
- |_ styles
    - |_ main.css
- |_ index.html

其中 images 文件夹用来放我们后面要用到的所有图片。styles 文件夹下的 main.css 里面用来写我们用到的所有 css 样式。最后 index.html 当然就是我们这次要写的页面了。

现在在 Notepad++ 里面新建一个文件，然后菜单 —— 格式 —— 以 **UTF-8** 格式编码，然后保存文件为 index.html。

网页乱码就是因为这里的编码没选对，现在不要再问我为什么乱码了哦～～

文件里粘入如下代码

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>我们的目标是抢前端的饭碗！</title>
        <link href="styles/main.css" rel="stylesheet" type="text/css">
    </head>
    <body>
        <!-- 后面的 Html 代码写这里，我就不老复制完整的网页代码了。 -->
    </body>
</html>
```

CSS 代码的建立方法一样，在里面粘贴如下内容：

```
/* 这是我写的第一个 CSS 文件，内心十分的激动，在这心潮澎湃之余，我想到了一个真理 —— 稻米鼠真帅！ */
```

认真写，不许错字，否则后果很严重！

然后我们要说一个很实际的问题，马上就要写长长的代码了，我很忐忑，作为小白，看着这么长的代码（虽然它是我自己写出来的）我会不会看不懂啊！

会，当然会了，我现在都懒得回头瞅我自己写的代码呢......所以我们要一边写代码，一边给代码加上一些注释进行说明，好让自己再回头看的时候不至于迷惑，这是一件很有必要的事情。

那么如何注释呢？在 Html 代码里注释的格式如下：

```
<!-- 这里的内容就是注释 -->
```

下面是 CSS 文件里的注释格式：

```
/* 这样写就是 css 代码中的注释 */
```

想想标记语言是怎么回事来着？现在我们就是用特定的符号来标记出来注释的内容。那么现在要记住：注释是为了我们日后能够方便的理解代码用的，除此之外，他不会起任何其他作用。他不会被显示，也不会对页面造成影响，你就当他是电影中的旁白好了。

虽然注释不起作用，但是你也不能给他乱放，否则至少也会让我们不方便阅读。一般我们新起一行书写注释或者把注释放在行尾。

好了，准备工作做完了，下节课开始写这个页面的导航部分～一个属于你的战役就此拉开了序幕......（我好像是讨人厌的旁白......）

> 本章学习卡片：卡片 **15**

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com** （刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发： http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书： http://www.jianshu.com/users/d37eaf3de0ff
- 站酷： http://www.zcool.com.cn/u/12927742

第十六章 导航条（一）

> 所有的读者都可以通过此链接（https://code.pubu.im/reg/b6mebkgg18lbtrr）注册，来和我们交流。
>
> 已付费的读者请与我联系，告知我您的支付宝账号与常用邮箱，我会邀请您加入 VIP 交流频道，方便我们更深入的探讨代码。

我就一直在想，我们应该把这个页面写成什么样的，是高大上还是......还好，后来我的懒惰打败了我——写个简单的！

但是不管怎么说，再懒也得动手了，先写个导航，嗯，虽然是个单页面网站，但是还是可以有导航的。反正也没老板指手画脚，所以随心所欲点挺好的。

其实说呢，写个导航很简单的，有啥啊？咱们合计合计啊，要个 Logo，几个导航项（什么首页啊，联系啊，关于啊这类的），一个搜索（前边我们学了主要功能代码），最多再加个登陆注册。这就是一般导航条（放这么多东西还叫导航条？）的所有内容。

我说啦，咱就一个单页面网站，一个 Logo 和一组导航项妥妥的了～。那我们就把他们写出来哦～

```
<div id="nav">
</div>
```

这段代码写在哪里知道吧，上一章的代码里我已经标识出来了哦，

然后我们同步的去写 CSS，就是在我们前边的 CSS 文件里写下如下代码：

```
#nav {
    width:100%;
    height:50px;
    background:#F3F3F3;
}
```

开始解释啊，因为这是第一次，我会努力说的很详细，以后会简单些。

`#nav` 是说 `id="nav"` 的元素，就是先说名字，再说事情，那个张二狗啊，今天的戏里你去扮演旺财，穿黄色的戏服，再戴上长耳朵哦。叫了名字，就告诉他，大括号里就是你今天的行头，都穿上就对了。

然后我们看看 `#nav` 今天都有什么行头哦。`width:100%;` 这个是说宽度，宽度 100%，这里就又有问题了，这是谁的 100% ？百分比是有参照物的，那么在这里的百分比是说父元素的百分之多少，什么是父元素？顾名思义就是他上一层的元素，换言之是套在他外边紧挨着的

那一层（不是套在外边就行，是紧挨着的！），所以 `#nav` 的父元素就是 `<body></body>` 。为什么不是 `<html></html>` ？虽然也是套在外面，但他不是紧挨着的，中间还套了一层 `<body></body>` 。所以， `<body></body>` 是父元素，而如果非要论起来 `<html></html>` 就是爷爷元素了（事实上没人这么说，也没这个名词，在此只是为了解释）。

现在 `#nav` 的宽度就和 `body` 的宽度一样了，那高度呢？我们写了 `50px` ，这个就没啥解释的了。最后 `background` 设置的是背景，背景色的值为 `#F3F3F3` （这个看不懂的朋友去研究一下 RGB 颜色的 HEX 格式），当然颜色也可以用 red，blue 之类颜色名称表示。

然后我们就可以看看效果了，如果你写对了的话基本应该是这个样子的：



来注意看灰色那块就是我们刚才写出来的效果。

然后你说了，看起来你想做的这个导航应该是通栏的，但是好像左右没到头，上边也差一点啊！嗯，我来解释一下，这不是我的另类的设计，这是我们写代码遇到的第一个问题，作为一个通栏导航，我当然希望他的上左右三面都紧贴着边的，可是现在没成功，为什么呢？

是因为浏览器对每个元素（标签）都有一个默认的样式（所以前边我们说 div 的默认样式几乎为零，所以很好用），那么根据我刚给出的这个条件来判断，虽然我们还不是很了解情况，但是也基本可以判断出问题更可能出在 body 身上。

确实的，这里是因为 body 默认有一个外补（margin）造成的，关于外补（margin）我们晚些再讲，现在先来解决问题，方法当然很简单，就是把外补（margin）设置为零就可以了。但是一般的一些谨小慎微，希望不出任何意外的朋友会把内补（padding）顺便也设置为 0 ，而且既然 body 设置了，为什么不顺便给 html 也来一发呢？所以 CSS 里面添加如下内容：

```
html, body {
    margin:0;
    padding:0;
}
```

大家注意了哦，上面代码中的 html 和 body 分别指的是两个标签，二者两个标签有相同的 CSS 属性，所以我们放在一起定义，我们用一个英文逗号来分隔两个选择器，然后后面大括号写上他们的共同属性，当然了更多个元素也是这样的格式去写，很方便是不是？就好像张三，李四你俩穿一样的行头！一个意思。

因为这个定义是最外层的，我们本着从整体到局部的原则，把他写在前边，而 `#nav` 的定义因为是内层，所以放在后面，这样符合逻辑顺序的 CSS 是为了方便我们日后的阅读，是一个习惯问题。

所以我们现在的 main.css 文件全文如下：

```
/* 这是我写的第一个 CSS 文件，内心十分的激动，在这心潮澎湃之余，我想到了一个真理 —— 稻米鼠真帅！ */
html, body {
    margin:0;
    padding:0;
}
#nav {
    width:100%;
    height:50px;
    background:#F3F3F3;
}
```

index.html 文件全文：

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8">
        <title>我们的目标是抢前端的饭碗！</title>
        <link href="styles/main.css" rel="stylesheet" type="text/css">
    </head>
    <body>
        <div id="nav">
        </div>
    </body>
</html>
```

来看看现在的效果哦：



终于三边都挨上了，欧耶～

那么大家都看到了，其实 CSS 文件就是这样一条一条的去定义元素的样式，我们只要知道这个元素有什么样式，和这个样式都可以有什么属性值就好。当然这些要靠我们的日常练习去积累，而不是死记硬背，一定要记住这一点。然后补充一些小知识哦～

> 在 CSS 里数值如果有单位就必须带上单位，否则会出错。但是有一个特殊情况是 0，因为无论什么单位，0 是等价的（幸好不用考虑摄氏度和华氏度），不会因为没有单位产生歧义。
>
> Html 里的属性值我们却常常省略单位，因为在 Html 里的属性值若不特别说明，则默认单位是 px（像素）

> 本章学习卡片：卡片 **16**
>
> 本章代码下载：本章代码

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com** （刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发： http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书： http://www.jianshu.com/users/d37eaf3de0ff
- 站酷： http://www.zcool.com.cn/u/12927742

第十七章 导航条（二）

现在我们来加 Logo 哦～，话说我这么懒就不用图片 Logo 了～。

代码如下，加在 #nav（就是 `id="nav"` 的元素） 里面：

```
<div id="logo">
    <a href="#">代码能有多难？</a>
</div>
```

挺简单易懂的，一个 div 里面放了个链接而已，你要是用图片就把 `代码能有多难？` 换成 `<img src="你的图片地址" />` 就行，这个大家应该是会的。

然后说说这个 `href="#"` 是个什么东东，他就是链接到当前页面本身，当然其实他是用了锚点，不过我们后面再说。然后这个链接要不要新窗口打开？当然不要，一个单页面网站点一次 Logo 复制一个标签页算什么神奇体验？所以关于是否新标签打开这个细节我们要多做思考，从实际体验和读者需求出发去做决定，而不是听老板的全部新标签打开的奇特理念。其实在日常我们用鼠标中键点击（滚轮也能按，别说你不知道）链接，就可以直接新标签打开，完全无视网页本身的设置，这才叫随心所欲的体验。可惜很多人不喜欢学习这些使用技巧......

然后我们来看看代码效果哦～



你说，妈呀！丑爆了，不跟你学了！

是挺丑的，本来不加链接还勉强可以看，这加上链接连我都看不下去了呢，要不咱们一起逃学？不过在走之前我们先把 CSS 写一下，也算善始善终了。

虽然我们是小白，但是也大概猜得出来这蓝色带着下划线的效果是链接造成的，而且我们说 div 几乎不带样式，而我们也没做什么定义呢，那就是链接的问题了。我们来写 CSS：

```
#logo a {
    color:#333;
    text-decoration: none;
    font-size:24px;
}
```

然后解释一下，先看选择器 `#logo a` 这是什么意思呢，是 `#logo` 元素里面的所有 `a` 元素，这里就是说 `id="logo"` 这个 div 里面的所有链接（a），所以明白啦，用空格分隔的的选择器就是什么里面的什么。就好像说财务部的会计们，下午给大家发年终奖！是指财务部里面的会计，其他的不要。这里是 `#logo` 里面的所有 `a` 们。你们的样式是这个样子的：

color 是颜色，确切的说是文字颜色，#333 其实是 #333333 的简写，当颜色值的数字两两相同的时候可以简写，比如：#CDF 就代表 #CCDDFF 。

text-decoration 是文字的下划线，这里写 none 意思就是没有啦，因为我们不想要下划线。

font-size 是字体大小，前边讲过，我们这里选择 24px。

然后我们再看一眼效果哦～



诶，好像不那么难看了耶，但是还有两个问题，要是能解决的话我们觉得还是可以继续学习下去的。第一个是你这在垂直方向上都不居中，往上飘着那么明显老板看到会骂死你的；第二个是能不能设置成粗体看看效果？

对于 CSS 来说，垂直居中是个比较麻烦的事情，他根据不同的情况有着多种的解决方案，在这里我们先来讲一种简单的哦。当你的内容是单行文本的时候，你给他设置一个行高，这个行高等于他外部元素的高度，那么文字在它的外部元素中垂直居中。

说简单点行高就是这一行文字的高度，你设置一个 30px 的行高，但是文字大小设置的是 12px，明显文字高度不够用，于是上下就要补充空白，而美好的事情是上下补的空白是一致的，那我们们就利用这个来做垂直居中了。

所以上边给加一个

```
line-height: 50px;
```

就搞定垂直居中的问题了，现在说加粗，这个属性很简单 `font-weight` ，然后你说属性值写 `bold` ，你看前面讲的单词我还记得哦～当然没问题，不过这个属性的属性值挺有意思的，所以我们在这里多说几句好了。

我们可以用 normal（普通），bold（粗体）这样的值对他进行定义，也可以用 100 —— 900 的整百的数字去定义字体的粗细，其中 400 相当于 normal，而 700 相当于 bold，明显的，我们用数字可以更细致的去设置，也更直观一些。那么用什么就看你个人的爱好了。

那么我们加入如下属性：

```
font-weight: 700;
```

然后看看效果：



虽然不算好看，但是我们这节课也算是想做什么就做出了什么，很随心所欲的样子！

> 本章学习卡片：卡片 **17**
>
> 本章代码下载：本章代码

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com**（刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发：http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/

- 简书：http://www.jianshu.com/users/d37eaf3de0ff
- 站酷：http://www.zcool.com.cn/u/12927742

第十八章 导航条（三）

我刚写下这个标题你就跑过来大喊着，终于又开课了，上节课我就有点觉得不对劲，你说那文字紧贴着左边能看吗？赶紧想想办法！

嗯，是的，紧贴着左边确实不好看啊，那我们这节课来想办法解决这个。这时候就要用到我们前边的盒模型了。那么我先开始打比方，打完比方再讲要用的一些属性，最后是解决今天的问题哦。

我们说 div 是一个盒子，就是个纸箱啊，方方正正的，四面有纸板挡着，可以装东西，也可以把内外的东西分隔开来，当然上面还可以摞上东西，或者底下垫上东西。就仿佛快递库房里的一只普通的不能再普通的纸箱，混迹于亿万同类之中，平凡，仿佛是他的唯一标签……（啊？！这不是在简书写散文？不好意思我跑题了）

当然了，其实这是我说的白话，但是我觉得很好的说明了他的特性，官方描述管这叫做容器，咱没那高的领悟能力，一说容器我就想起烧杯什么的……那就没法理解去了。你就知道这么个事就行，别回头一说起容器来不知道，然后再往下说就说是看稻米鼠那个扯淡的教程学的……

我就觉着快递箱这个比喻太方便，太形象了，所以我坚持这么叫了。咱们继续说快递箱，然后我们开始想象，页面是一个大屋子，我们从左上角开始往下一个挨一个的码纸箱，就码一层，那肯定就按着顺序排过来了，这连小孩子过家家都会玩。这个叫做文档流，别管他啥意思，记住这个比方和对应的名词就 OK ，以后我们就可以交流。我总得告诉你这是扳手，那是锤子，以后着急的时候就可以跟你吼：“小心锤子别砸了你的手！”，一下都明白，否则我跟你喊：“小心那个一头大一头小，丁字形的，用来砸东西的，铁的硬邦邦的玩意，别砸……”你已经砸完了，但至于他为什么叫锤子，你管呢！

前边我们做了一个 #nav 的快递箱，还做了一个 #logo 的快递箱。都设置了他们的尺寸，一个宽度，一个高度。这就是设置了两个快递箱的尺寸啊。一个长一个宽，别问我厚度，这里用不着。然后我们想啊，#nav 我们设置了一个背景，但是他里面装的东西（另一个箱子 #logo）我们却可以看到，这说明什么？我们设置的背景是箱子底的颜色，而箱子的顶是透明的，你想想是不是？顶不透明我们就看不到里边了啊，背景色又填满了整个箱子，却遮不住箱子里的东西，那它就是底面的颜色喽。

好了，我们再考虑啊，你说 #logo 我们没设置背景吧，可是为啥底下不是白色啥的？确是套在他外边的箱子的背景色（底面颜色）？这只能说明一个问题，div 默认的背景色是透明的，也就是这个快递箱啊默认顶面和底面都是透明的，然后我们看到另外四个面（上下左右）了么？，好了他们也是透明的！

然后我们再想大家都收过快递，易碎品什么的是只用一个纸箱装了就行么？不是吧，总是要做防护措施的，防护措施不外乎两种：1、 箱子里面加上填充物；2、箱子外面裹上汽泡纸。那么填充物使得我们的货物需要一个比他原本体积更大的箱子，将将好装下货物的箱子还怎么放填充物，必须换大箱子，结果你收到快递，呀！这东西这么大？结果是里边放了填充物。那么这个填充物就是内补，在箱子里边补充一下嘛～

然后我们再说外面裹的泡泡纸，裹上泡泡纸会影响箱子的大小么？不会，箱子还是那么大，但是裹上泡泡纸他们整体占得面积就大了，这个纸箱和那个纸箱之间就产生了距离，这个距离是多少？就是一个的泡泡纸厚度加上另一个泡泡纸的厚度。对吧？这就是外补。

再想，假如无论填充物还是泡泡纸，都是透明的，就是看不到啊，当然东西还是存在的。那么我们加了填充物（内补）之后我们看到的是什么？货物和箱子之间保持着一定的距离，对吧，因为有看不见的填充物在那里呢。加了泡泡纸（外补），我们看到什么？箱子之间产生了距离，因为看不见的泡泡纸在他们之间呢。

这样我们就理解了外补和内补。我们说箱子的底面可以定义颜色，那么周围的四面（上下左右）可不可以定义呢？可以啊，可以定义颜色还可以定义厚度呢，原本这个箱子的四壁是没有厚度的，存在，但是厚度是 0 ，能理解吧，就是文学中的薄若蝉翼的意思，有，但是看不到，因为太薄了，还是透明的。所以当我们设置了它的厚度和颜色我们就可以看到他了吧？这就是边框。

那么现在这个模型建立起来了，一个盒子从外往里依次是：

> 泡泡纸 —— 外补
>
> 纸箱壁 —— 边框
>
> 填充物 —— 内补
>
> 易碎品 —— 内容

到这里都很好理解吧，说不好理解的那个去首富家多买点东西就懂了。

这比方打完了，你可得记住，这是理解后面的关键哦！我从来不说代码是关健，你理解他才是关键，代码是可以查的，但是怎么理解他，然后能用他，这是你该掌握的。

然后说 CSS 属性：

> 内补：padding
>
> 外补：margin
>
> 边框：border

这三个属性有些共同点，比如你给他们赋予一个数值，那就是四边都一样，都是这个数值，这很好理解。平均主义，你有我又大家有。

但是事实上他们都涉及了四个方向，其实四个方向可以分别设置的，那么我们给四个数值的话，比如： `margin ：1px 2px 3px 4px;` 看到啦，用空格分隔数值，这四个数值怎么对应方向的呢？记住啊，页面是从上开始写，所以我们从上面开始，顺时针旋转，于是就是：上、右、下、左。

这也很好理解，按着顺时针顺序挨个赋值而已，但是要是遇到变态呢，我就写了俩数值，那就是第一个指的是上边和下边都是这个数值，第二个值指的是左边和右边都是这个值。这个是平均分配外加面对面分果果......反正总而言之还是可以理解的。

但是我要是就给三个数值呢？？？

太变态了！挨个分，上边，右边，下边，都有了，左边问啊："那我呢？那我呢？"你也不好意思说没他的份啊，所以就让他跟对面的共用一个数值算了。所以三个值代表：上、左右、下。

然后呢，其实这三个属性都是简写属性，就是把几个属性合在一起写的。要是分开写呢？比如：

```
margin-top:1px;
margin-right:2px;
margin-bottom:3px;
margin-left:4px;
```

四个方向可以这样分别定义，上面的四个方向分别是上右下左。当然，border 的定义要略复杂一些，因为他不光需要一个宽度，还需要边框的样式和边框颜色的定义。这些我们留在以后再说。

现在在回到本章的开头，想想我们其实是要做什么呢？两边留上一些空白，对吧，但是这些空白还要有背景，其实就是在 #nav 元素的里边对不对？那就是给 #nav 加内补喽～

你说好像我给 #logo 加上外补也能达到相同的效果，但是你忽略了一个隐含条件：我们说左边需要空白，其实如果右边有内容的话你会说右边也需要留出空白的，而加在 #logo 上的属性显然不能满足这一要求。只有给 #nav 这个盒子左右两侧放上填充物才能比较针对性的解决上述问题。

好了，我们废话了将近三千字，只是为了得出一个结论，我们应该给 #nav 加上如下属性：

```
padding:0 30px;
```

> 本章学习卡片：**卡片 18**
>
> 本章代码下载：本章代码

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com**（刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发：http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书：http://www.jianshu.com/users/d37eaf3de0ff
- 站酷：http://www.zcool.com.cn/u/12927742

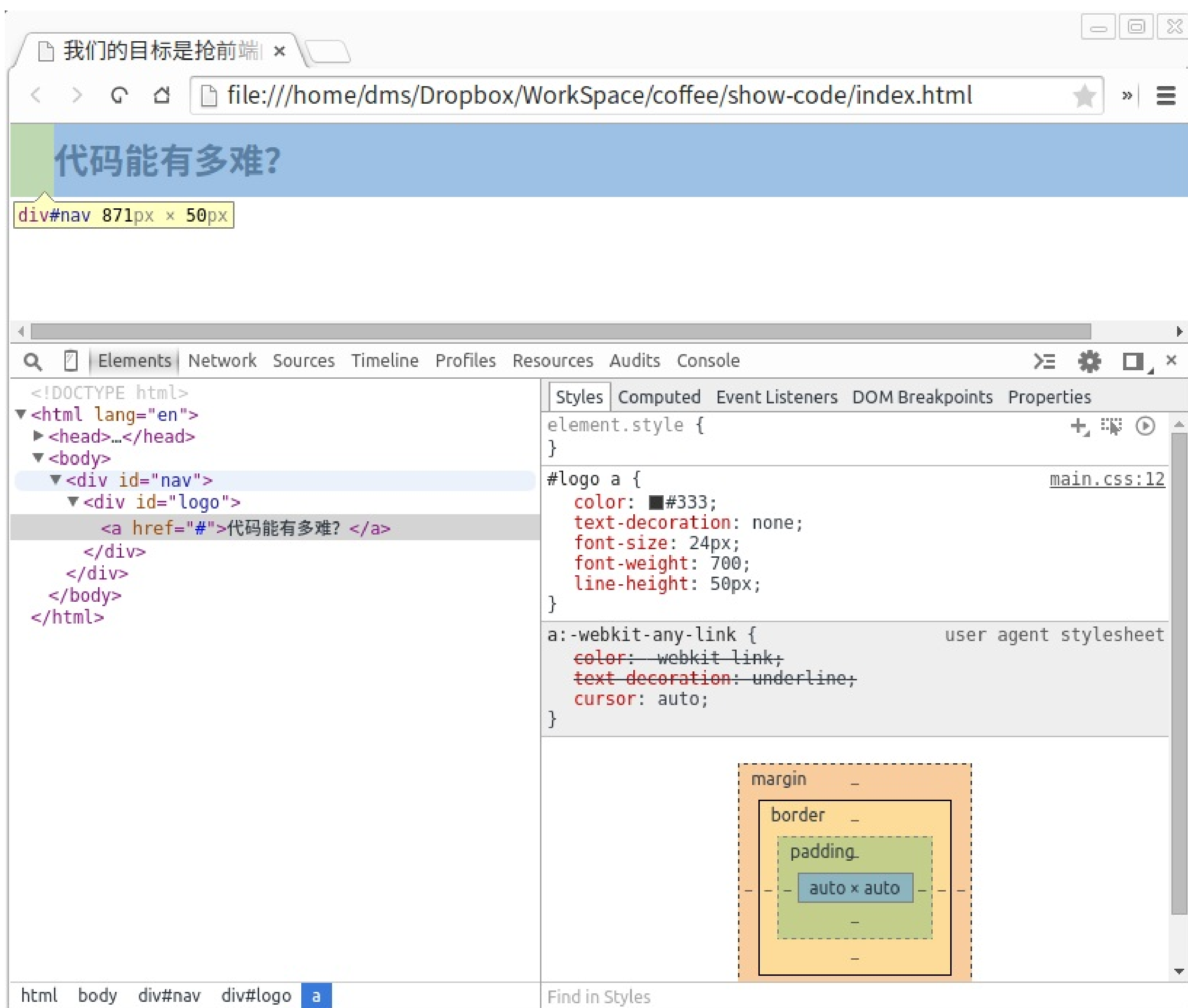
第十九章 导航条（四）

你说，上节课的都听明白了，就是不懂。

这很正常，现在我们来说一个你以后在网页前端道路上不可或缺的助手——“审查元素”（这是 Chrome 浏览器里的名字，而 Firefox 里叫做“查看元素”）。

其实很简单，你对着你要研究的元素右键，然后选择审查元素就行了。

然后我们就看到一个如下图的界面（Firefox 中类似）：



左下方是网页当前用来显示的代码，先记住我这个说法，到后面我们研究 JavaScript 的时候你就明白我为什么这么说了。

现在深灰色的那行就是我们刚才右键的元素所对应的代码。而右侧上边的内容就是这个元素所对应的 CSS 代码（只是与这个元素有关的部分哦）。

往下看，右下方那个黄色绿色蓝色一层套一层的方块，就是当前元素的盒模型了，结合上节课的内容，其实你稍微认真看下就能理解了。

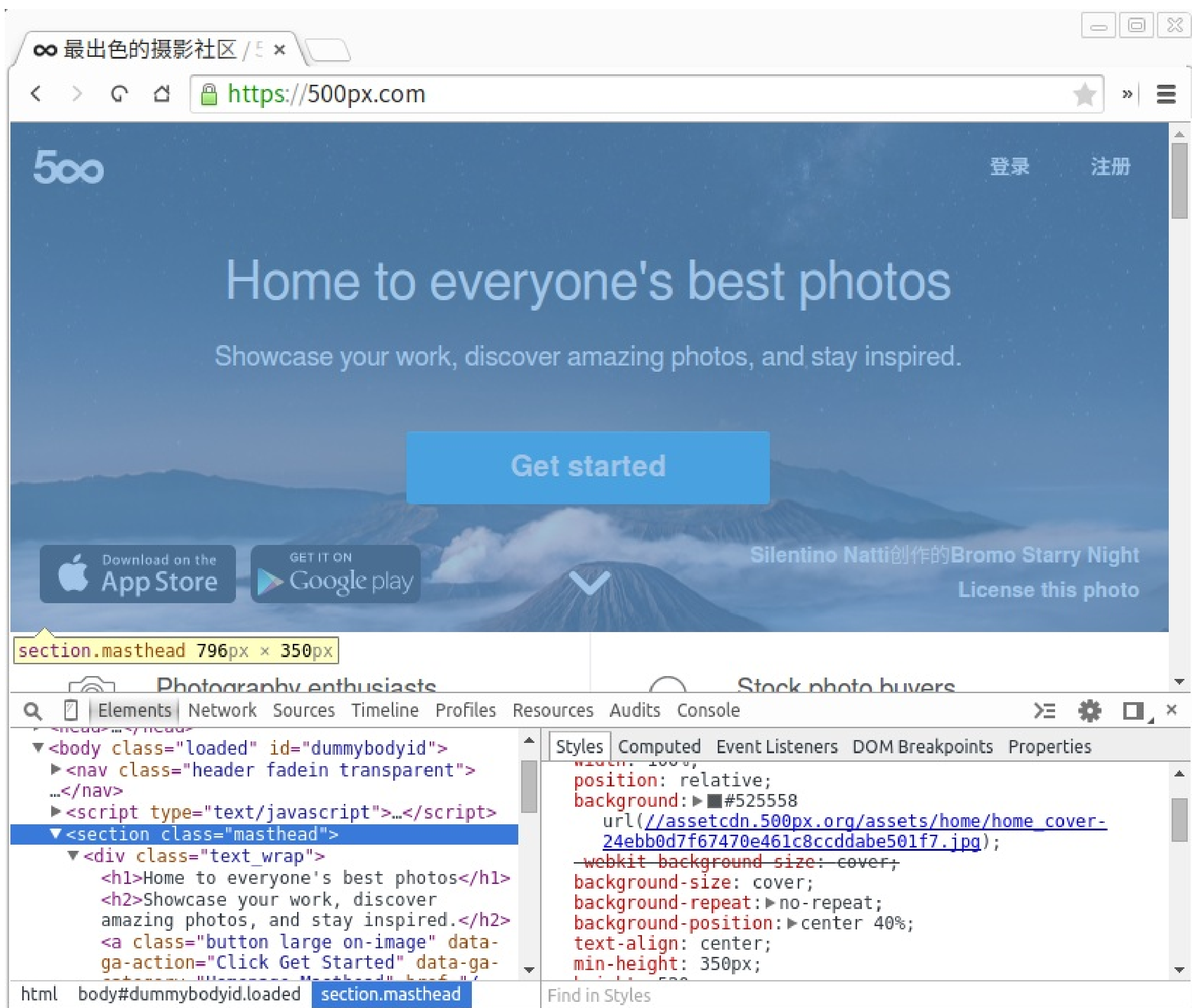
然后如果你觉得 CSS 一套一套的看着眼晕，就点击那个 computed（计算后） 标签，意思就是把重复的合并，未生效的省略……之后，当前元素实际的有效的 CSS 属性和对应的值。非常方便。

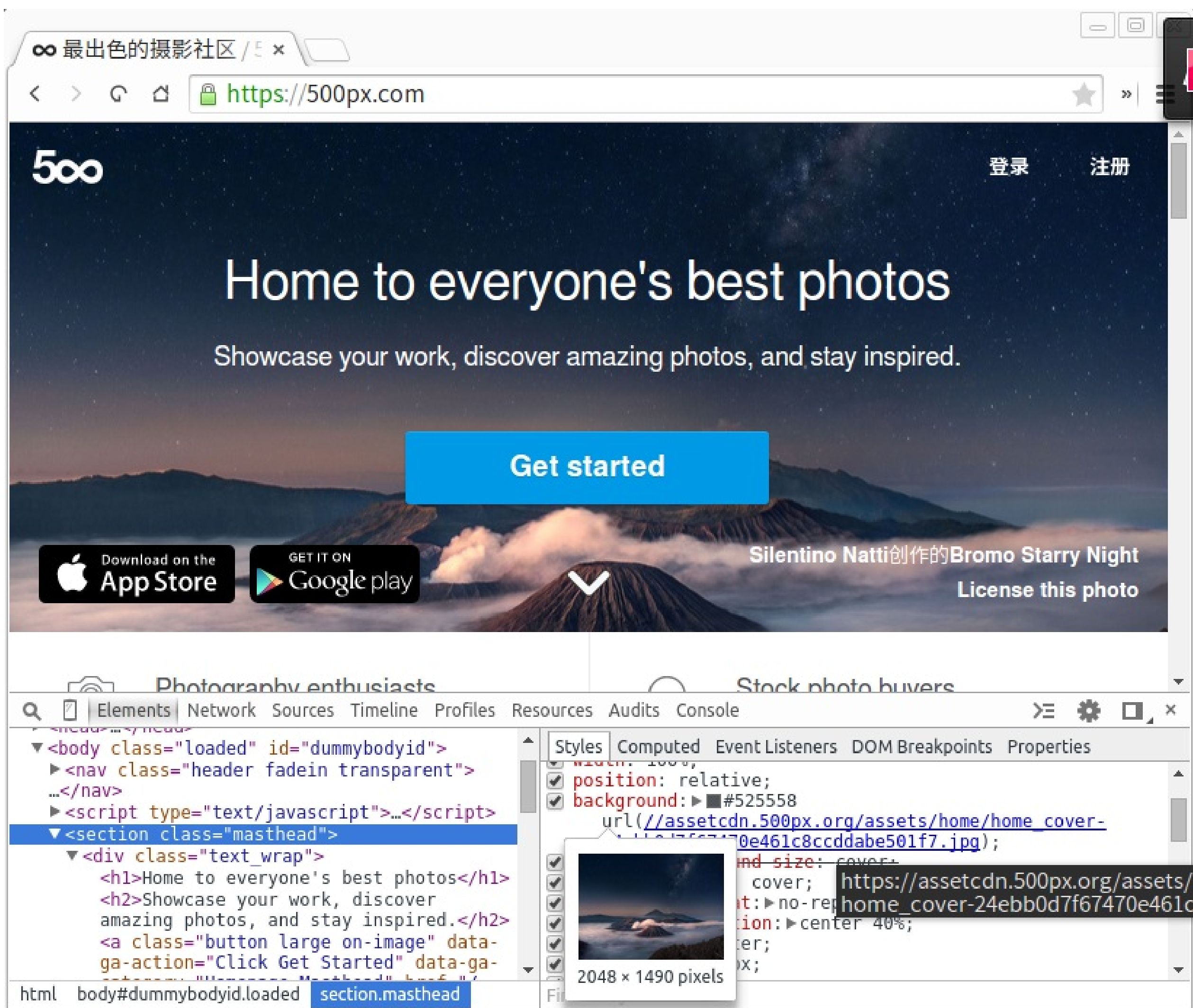
我们再回到左边，用鼠标在代码上滑动（不是点击，也不是拖动），你会看到一个蓝色的条着重显示你鼠标指向的代码。同时上面网页上的某些元素会被用特殊的颜色标识出来。所以现在你指哪行代码，那行代码所对应的元素就会在页面上标记出来，而且你注意他的标识颜色，与右下方盒模型的颜色是对应的。这样哪里是外补哪里是内补就一目了然了。

然后说说那些小图标，左上那个小手机的标志是移动设备视图，现在我们还用不上，将来我们讲响应式网页的时候会用到。右边齿轮是设置，叉是关闭。那个一半黑一半白的图标是说现在这个面板所处的位置，你可以点击它调整位置看看效果，反正再找相应的图标点击还能回来，怎么方便怎么用就是了。

这个简单吧，你看我就这么几句话，你还都能听得懂，就把我们眼下用的到的都讲清楚了。来给你讲个案例让你了解他的强大。

我们知道背景图片是不可以右键另存为的，但是有的时候图片真的很美，怎么把他们搞下来呢？我们打开 500px 来做实验，在它的首页大图上右键，然后鼠标在代码上上下滑动找那行可以让蓝色正好覆盖整整个图片的代码，其实也就是说这个元素的内容部分正好和图片一致，点击这一行，然后看右侧的 css ，上下滚动寻找 background 后面的网址，鼠标放上去看看，会有图片缩略图，基本就是这张图片了。如果没找到，那么在上一层下一层元素查查也基本差不多了。


最出色的摄影社区
https://500px.com
500
登录
注册
Home to everyone's best photos
Showcase your work, discover amazing photos, and stay inspired.
Get started
Download on the App Store
GET IT ON Google play
Silentino Natti创作的Bromo Starry Night
License this photo
section.masthead 796px × 350px
Elements Network Sources Timeline Profiles Resources Audits Console
<body class="loaded" id="dummybodyid">
<nav class="header fadein transparent">…</nav>
<script type="text/javascript">…</script>
<section class="masthead">
<div class="text_wrap">
<h1>Home to everyone's best photos</h1>
<h2>Showcase your work, discover amazing photos, and stay inspired.</h2>
<a class="button large on-image" data-ga-action="Click Get Started" data-ga-
Styles Computed Event Listeners DOM Breakpoints Properties
position: relative;
background: #525558
url(//assetcdn.500px.org/assets/home/home_cover-24ebb0d7f67470e461c8ccddabe501f7.jpg);
-webkit-background-size: cover;
background-size: cover;
background-repeat: no-repeat;
background-position: center 40%;
text-align: center;
min-height: 350px;
html body#dummybodyid.loaded section.masthead
Find in Styles



你说页面不允许右键？那你按一下 F12 键试试看。当然了，在这里还是要强调一下，技术无罪，但你得用来干正事，反正白茶同学说过：

> 长期盗图，对投胎不好！

顺便说一句，如果你在我下方所列出的四个站点之外看到本教程，基本上他们是盗版的！！！

好了，回归主题，有一句话说能看到就能改！（来自某玄幻小说）用来形容这个工具还是十分恰当的，反正那些代码想改哪里双击就是了（你别很实在的双击里面链接就好）。然后尽情的修改和测试吧～～

这东西实在太过强大和有用，建议每天玩一会，玩懂他！

> 本章学习卡片：卡片 **19**

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com** （刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发： http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书： http://www.jianshu.com/users/d37eaf3de0ff
- 站酷： http://www.zcool.com.cn/u/12927742

第二十章 导航条（五）

> 如果本章演示效果无法正常展示，请至主站查看 http://coffee.zji.me/

现在来写导航项。其实导航是有专门的标签的 `<nav> </nav>` 但是这个特性比较新，IE9 才开始支持，所以很多人还是习惯用列表来写导航，这里我们也使用列表，为的是你以后看到各种代码都能看得懂，因为 nav 标签相对于列表还是比较简单的。

那么先来说说列表哦：

```
<ul>
    <li>列表内容</li>
    <li>列表内容</li>
    <li>列表内容</li>
</ul>
```

效果如下：

- 列表内容
- 列表内容
- 列表内容

如上叫做无序列表。

```
<ol>
    <li>列表内容</li>
    <li>列表内容</li>
    <li>列表内容</li>
</ol>
```

效果如下：

1. 列表内容
2. 列表内容
3. 列表内容

如上叫做有序列表。

你说这跟导航项一点都不像啊，但是在逻辑上还是很像的，都是列出来一些内容。既然我们对于前边的编号圆点什么的完全不在乎，那么我们就选择无序列表好了，那么 html 代码很简单的，几乎跟上面一样

```
<ul id="nav-items">
    <li class="nav-item">首页</li>
    <li class="nav-item">下载</li>
    <li class="nav-item">特点</li>
    <li class="nav-item">关于</li>
</ul>
```

这个样子没有问题，挺容易看懂的，我们把他加在 #loge 的 div 后面来看看效果。



你说，哼，幸亏我已经不再相信你了，否则又得失望。本来想要跟 logo 在一排的效果（要不那背景留着干嘛用？？）结果又做出来一个奇葩的造型。

我跟你说啊：想横排，加浮动！你看我一步步把它搞上去哦！在 CSS 里加上

```
#nav-items .nav-item {
    float:left;
}
```

就是给我们的每一个导航项加上了一个向左浮动的属性，我们只做了这一点，来看看什么变化哦



这个......好像是到一行了，但是有点乱，看不好效果，主要是那个圆点的影响，那我们去掉圆点，把刚才那部分 CSS 再加上一个属性：

```
#nav-items .nav-item {
    float:left;
    list-style: none;
}
```

就是列表的样式：无。再看效果。



呀，果然是一排相邻的元素，一加上浮动就从竖着排列变到横着排了。那么眼下呢，到这里你只要记住把想要横排的东西都加上浮动就好。如果你想知道为什么，那我们需要继续一下我们快递箱的故事，有些绕，这个故事晚些时候我会发布在我们交流平台的“咖啡俱乐部”频道内（就是我们的 VIP 频道）。

现在我们就先记住这个结论，然后再实践中去慢慢体验他的用途。然后我们做个练习，你看 logo 和导航项这两部分还是纵向排列的呢，那我们来给他俩也加上浮动试试看。

为了防止你搞乱了，我在这贴一下完整的 CSS 代码：

```
/* 这是我写的第一个 CSS 文件，内心十分的激动，在这心潮澎湃之余，我想到了一个真理 —— 稻米鼠真帅！ */
html, body {
    margin:0;
    padding:0;
}
#nav {
    width:100%;
    height:50px;
    background:#F3F3F3;
    padding:0 30px;
}
#logo {
    float: left;
}
#logo a {
    color:#333;
    text-decoration: none;
    font-size:24px;
    font-weight: 700;
    line-height: 50px;
}
#nav-items {
    float: left;
}
#nav-items .nav-item {
    float:left;
    list-style: none;
}
```

再来看看效果：



那么到了这里貌似虽然难看些，但是我们把位置都搞对了。下节课我们来试着给导航项加链接哦～～

> 本章学习卡片：卡片 **20**
>
> 本章代码下载：本章代码

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com**（刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发： http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书： http://www.jianshu.com/users/d37eaf3de0ff
- 站酷： http://www.zcool.com.cn/u/12927742

第二十一章 导航条（六）

上次我们把导航条写了一个雏形，就是看起来还行（这么难看居然说还行......我眼瞎）。我们的计划是现在给它加上链接，让他成为一个能干活的导航。

加链接这种事情对于现在的我们来说没什么难度了吧。如果你坚持做练习的话一定觉得小事一桩了。但是我们现在遇到一个问题：我们要给谁加链接？

这还有疑问吗？有！我要不说肯定会有人这么写：

```
<a href="#"><li>这是错误的写法</li></a>
```

为啥错啦，因为你女盆友用的 IE6 浏览器不支持，这个理由够用吧～

可是你不依不饶啊，那为啥错了？你先记着，a 标签里面的元素只能是文字和图片哦，后面我们再去慢慢研究他。

现在我们把这个修改一下，于是我们的导航代码变成了：

```
<ul id="nav-items">
    <li class="nav-item"><a href="#">首页</a></li>
    <li class="nav-item"><a href="#download">下载</a></li>
    <li class="nav-item"><a href="#feature">特点</a></li>
    <li class="nav-item"><a href="#about">关于</a></li>
</ul>
```

大家说，你这链接地址好奇特，咋不是个网址啊，咱说了，这个问题留到后面处理，先别着急。咱们先来看看视觉效果。



你说你开始怀疑人生，至少是怀疑我的教程。你等等，你究竟觉得有啥不好的？文字的颜色和下划线的问题我们讲过怎么解决了吧？

看看前边的这部分 CSS ：

```
#logo a {
    color:#333;
    text-decoration: none;
    font-size:24px;
    font-weight: 700;
    line-height: 50px;
}
```

把它复制一下，修改成：

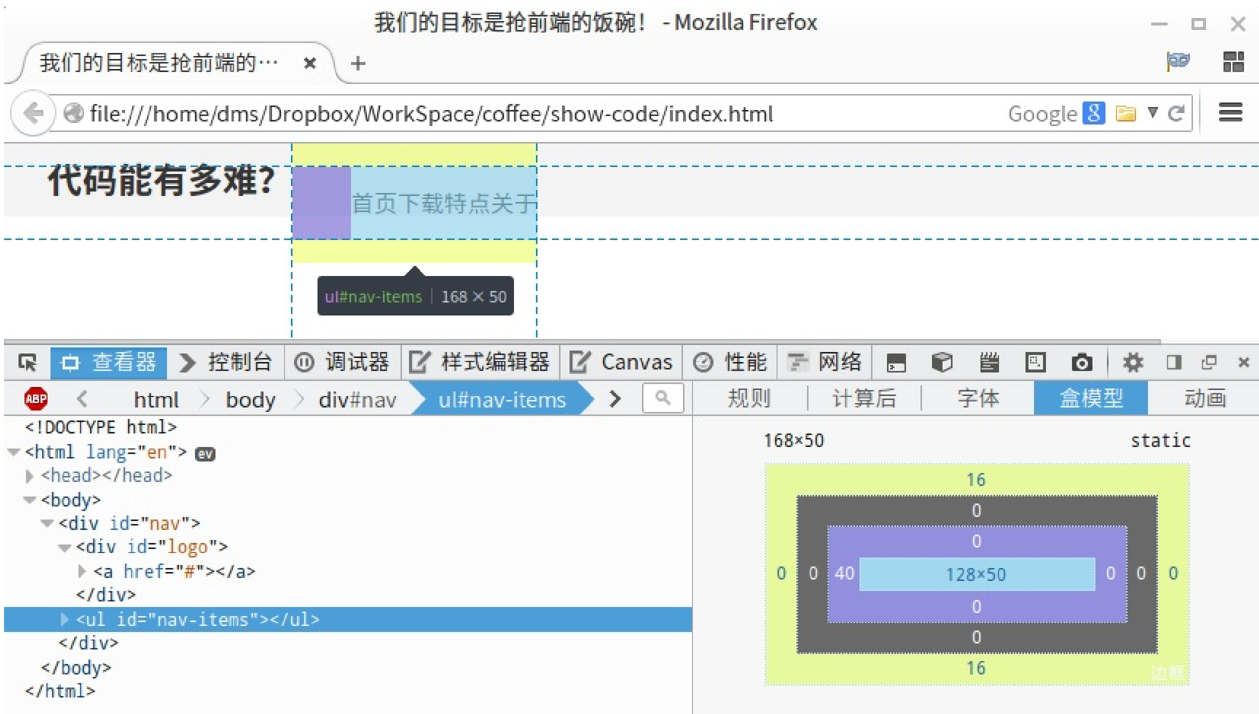
```
#nav-items .nav-item a {
    color:#333;
    text-decoration: none;
    font-size:16px;
    line-height: 50px;
}
```

注意选择器，现在是三层选择器了，下边的属性都看得懂了，因为前边说过了，那么来看看效果哦：



文字样式终于看的过去了，但是这位置......现在我们开始正式研究这个问题了哦～～强迫症的同学已经期待已久了吧？（嘿，别拍我，尊师重道！）

右键“审查元素”（Firefox 里是“查看元素”，以后我不说了），来看看 #nav-items 这个元素哦～

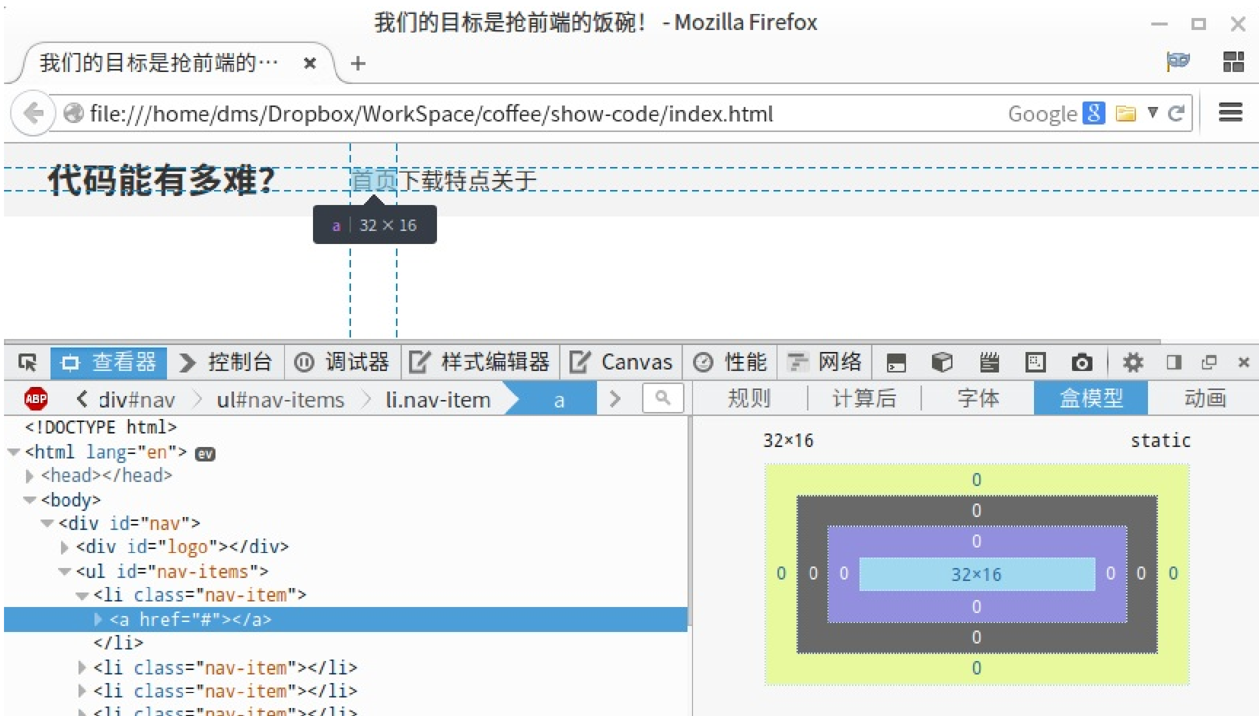


看到啦，他左侧有个 40px 的内补（图中蓝紫色的位置），上下各有 16px 的外补（图中黄色位置）。你望着左上角问我："哪有数值？"来，转头，看看右下角的盒模型。

这些属性是我们设置的吗？我们查了一下 CSS 文件，发现不是，那么他又是浏览器给的默认属性。这些默认属性是我们想要的吗？内补好像有用，因为我们的导航项要与 logo 拉开一些距离。但是上下的外补明显不是我们想要的。所以去掉他。

```
#nav-items {
    margin:0;
}
```

看看 CSS 里已经定义过 #nav-items 了，所以给他的大括号里追加上 `margin:0;` 就好了。然后来看下一个问题：



现在位置没问题了，但是你看每个导航项的可点击区域（就是你可以点击到这个链接的范围）才这么大（上图蓝色的区域），你觉得体验会好么？请各位读者瞄准了点击，否则后果自负～～

这肯定是不行的，你说，好办，给 a 加上宽高，你可以试试看，我猜他不行啊，为啥不行？在这里我们要引入一个新的属性了。`display`，显示，这个元素怎么显示呢？

先说一下他的常用属性值：

> none 此元素不会被显示。
>
> block 此元素将显示为块级元素，此元素前后会带有换行符。
>
> inline 默认。此元素会被显示为内联元素，元素前后没有换行符。

果然除了第一句，剩下的我也没看懂呢，还是用我的话来说吧，block 是个块，就像 div ，是个已经打包好了的箱子，很乖，说什么听什么，好处理；inline，字面是在行里边，就是跟文字一样的特性，没打包的一堆东西，跑来跑去的，不很听话，你跟他说宽高内补外补的他跟你假装没听懂。剩下的问题咱们在使用中慢慢体会就好。

我们想想啊，一段文字加上链接，如果一行没显示下，是不是会分成两行显示？这个凭经验我们也知道——是的。于是链接跟文字特性很像，那他就是内联元素（inline），所以就解释了为啥给他宽高他不听话了。可是不听话不行啊，咱们要让他听话的，所以要把他设置为 block，所以给 a 填上相应的属性：

```
#nav-items .nav-item a {
    color:#333;
    text-decoration: none;
    font-size:16px;
    line-height: 50px;
    display: block;
    height: 50px;
}
```

顺便高度 50px 也没有异议，那么宽度......你想过没有，要是出来一个导航项是五个字，你把它和两个字的导航项设置一样的宽度，他能好看吗？所以实际上我们要设置的是什么？是文字之间的间隔，对不对？只要两个导航项之间的文字间隔是一致的，就会显得很协调。这肯定就是内补或者外补了，现在的问题就是：加给谁？加什么？

告诉你啊，还加给 a ，加内补，为啥呢？无论是内补还是外补，无论是加给 a 还是加给 li 在外观上都是没啥区别的。但是 a 的可点击范围有区别。你说对对，还有这一茬呢，那加给 a 就对了，为啥是内补？简言之对我们点击进行反应的是箱子，而与箱子外面的泡泡纸无关，外补不算箱子的范围，但是内补算。所以我们给一个巧克力豆周边加上大量的填充物，然后弄个大箱子装就可以说是一大箱巧克力豆了。但是要是一个小盒子，装上巧克力豆再裹上半米泡泡纸，你说是一大箱就说不过去。现在需要箱子有一定的体积就往里塞喽～

```
#nav-items .nav-item a {
    color:#333;
    text-decoration: none;
    font-size:16px;
    line-height: 50px;
    display: block;
    height: 50px;
    padding: 0 20px;
}
```

再来看看效果：



是不是看起来舒服多了？下一节我们继续......写导航。

> 特别补充说明：本教程目前的 CSS 书写并不十分复合规范。
>
> 不是本人不知道规范，而是为了便于理解循序渐进，
>
> 现在是根据需求直接追加，只求达到效果，暂时不考虑细节。
>
> 等后面讲的差不多了再作规范，要不然都没认识几个 CSS 属性，我说谁先谁后你们也搞不明白。

> 本章学习卡片：卡片 **21**
>
> 本章代码下载：本章代码

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com**（刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

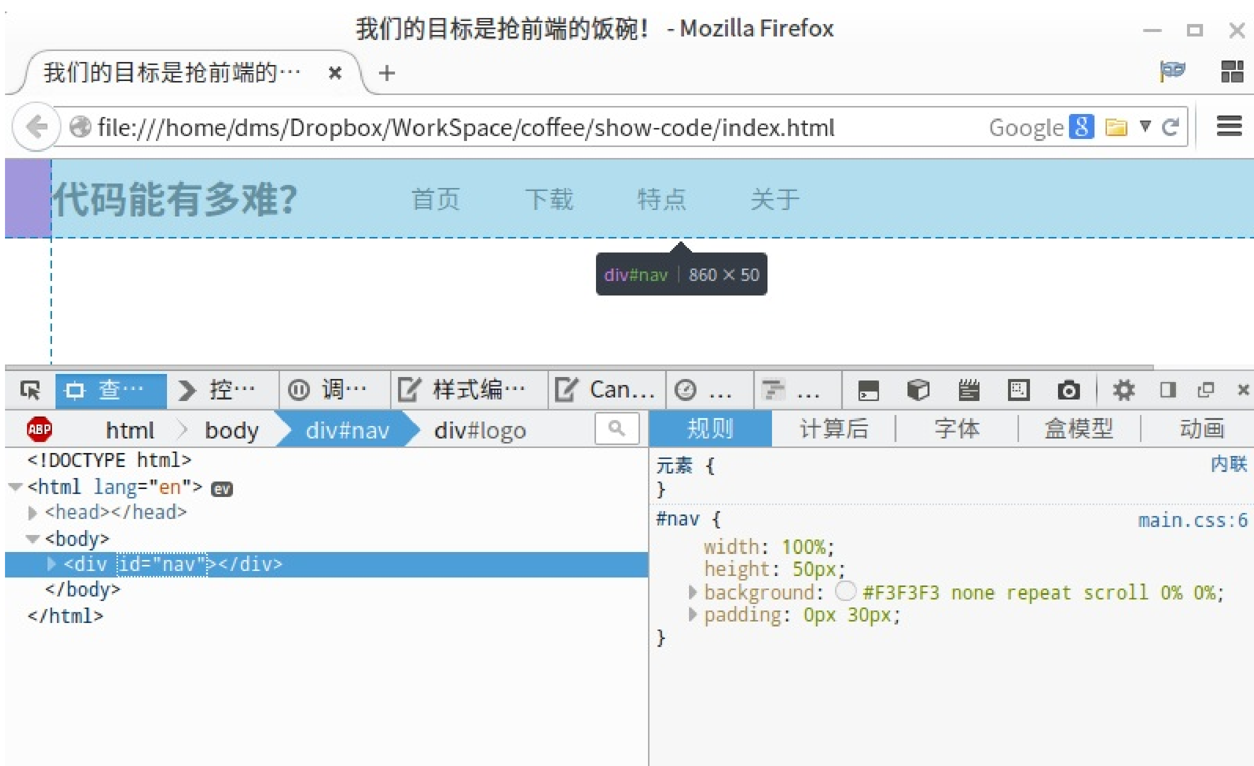
- 主站首发： http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书： http://www.jianshu.com/users/d37eaf3de0ff
- 站酷： http://www.zcool.com.cn/u/12927742

第二十二章 导航条（七）

现在我们来学着解决问题哦，我们还有一个遗留问题没有解决，就是页面底下的滚动条。你肯定又说了：“你还是知道解决啊？你不知道我都看着碍眼好多章了么？”这个不怨我啊，你们看看我的截图，我的系统是超细滚动条，所以我看着一点也不碍眼啊！（好了，可以扔砖头了，我好捡回家盖房子娶老婆......）

方法很简单啊，右键审查元素，然后鼠标在代码里划过每个元素，看看最可能是哪个元素引起的问题。

然后我就觉得最可能是这里出了问题，为啥啊，这么多元素，就它的宽度比我窗口宽度还大（我当前窗口的宽度为 800px）。



你看，其实查找问题也未必很难，那么究竟是出了什么问题呢？这里我们猜猜啊，我们知道他如果遵循我们设置的宽度的话，那么他的宽度应该是 100% ，当前也就是 800px（body 元素当前的宽度值）。再看看我们加的内补，左右各 30px。这要是 `800+30+30=860` ，正好是当前元素的宽度。这应该就是问题的原因了。

那么这里涉及到两个问题。第一个，我们设置的宽高到底是什么？其实我们设置的宽高，不是元素边框（边界）的宽高，倒更像是元素内可用面积的宽高。就好像我们说的不是快递箱的尺寸，是他加上填充物之后还能装多大面积的东西，理解没？

所以元素的实际尺寸（从边框开始测量），实际上是设置的宽高加上内补的尺寸。就是快递箱的尺寸等于里面的东西再加上填充物的尺寸。说简单点就是箱子里填充物（内补，**padding**）的多少会影响箱子的大小（元素的实际面积），当然外面裹多少泡泡纸（外补，**margin**）都不会影响箱子的大小。

这就理解了，那么内补我们并不想去掉，因为想着在左右留点空白用的。看来只能动宽度了。那宽度应该改作多少？其实去掉就可以了，因为块元素**(display:block)**不浮动的情况下默认填满一行。（这是第二个问题，详见 VIP 章节之 关于浮动的故事）

于是，去掉了 #nav 的宽度之后，我们获得了一个虽然简陋，但是确实看得过去的导航条。



回想我们学了这么多节课，其实主要在说一些基本的 CSS 知识了，真正看我们写下的代码却没有几行，再看看代码，差不多也能看懂一二了，好像也不过如此。

这是一个完整的导航了？当然不是，不过毕竟从无到有，从难看到现在看的过去，我们完成了一轮创造的过程，算得上是我们学习过程中一个重要里程碑！

如果到这里你基本都明白了，那么我来带着你做点坏事哦，挺简单的，我们把当前的 #nav 部分的 Html 代码复制一遍，修改一下 id（id 不能重复啊，而且我们也需要用他进行一些区分），这么说有点抽象，来看一下我改过之后的代码（我只粘贴 body 标签之间的喽～）

```
<div id="nav">
    <div id="logo">
    <a href="#">代码能有多难？</a>
    </div>
    <ul id="nav-items">
        <li class="nav-item"><a href="#">首页</a></li>
        <li class="nav-item"><a href="#download">下载</a></li>
        <li class="nav-item"><a href="#feature">特点</a></li>
        <li class="nav-item"><a href="#about">关于</a></li>
    </ul>
    <ul id="nav-items-r">
        <li class="nav-item"><a href="#signin">登录</a></li>
        <li class="nav-item"><a href="#signup">注册</a></li>
    </ul>
</div>
```

第二个 ul 标签部分是新加入的内容，这挺常用的哈，虽然其实这次我们用不到，但是玩玩还是无妨的。再简单修改一下 CSS，下面是原来的 CSS 中的一部分

```
#nav-items {
    float: left;
    margin:0;
}
#nav-items .nav-item {
    float:left;
    list-style: none;
}
#nav-items .nav-item a {
    color:#333;
    text-decoration: none;
    font-size:16px;
    line-height: 50px;
    display: block;
    height: 50px;
    padding: 0 20px;
}
```

现在第一条 `#nav-items` 不用动，但是我们要照猫画虎的给新添加的 `#nav-items-r` 加一条，只是改作向右浮动。下边两条现在明显只对第一个 ul 里的内容起作用，但是我想应用到整个 `#nav` 部分，所以把选择器中的 `#nav-items` 换成 `#nav` ，来看修改后的结果：

```
#nav-items {
    float: left;
    margin:0;
}
#nav-items-r {
    float: right;
    margin:0;
}
#nav .nav-item {
    float:left;
    list-style: none;
}
#nav .nav-item a {
    color:#333;
    text-decoration: none;
    font-size:16px;
    line-height: 50px;
    display: block;
    height: 50px;
    padding: 0 20px;
}
```

去看看效果吧～



于是我们得到了一个有左右两部分导航的导航条，现在我希望大家回过头去对整个导航部分认真的复习一下，免得学到后面把前边的忘记了，那样就容易混淆了。另外一定记得每一节课都要跟着做哦～～

下一节课我们来美化我们的导航条。

> 本章学习卡片：卡片 **22**
>
> 本章代码下载：本章代码

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com** （刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发： http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书： http://www.jianshu.com/users/d37eaf3de0ff
- 站酷： http://www.zcool.com.cn/u/12927742

第二十三章 导航条（八）

我们来做一些简单的美化工作，本着过犹不及的原则（其实是我懒），我们就做点很简单的东西啊，比如：你看两个导航项之间的分隔不明确是吧，我们来加个分割线啊，挺简单的事，给 `#nav .nav-item` 选择器下加上 `border-right:1px solid #DDD;` 就可以了，就是每个导航项的右侧边框是一像素粗细，实线（solid），颜色是 #DDD。看到啦，边框就这么设置，简单吧～

然后为什么只设置右边的？大家想想就明白啦，每个导航项的左边都和另一个导航项共用一条线啊，这个可以理解吧。但是这里还是多了一个小麻烦哦，就是每组导航项的最后一个，他右侧的边框没人来共用，孤零零，显得好多余啊～

所以加分割不难，而去掉多余的分隔却有点麻烦。这里有两种方法可以去掉分隔，但是在这之前我想我要先说一下 CSS 的优先顺序。

首先我们知道浏览器会给某些元素默认的样式，这个优先级最低，因为你做的任何与之冲突的自定义都会将它覆盖掉。你看前边我们遇到默认样式的问题就直接再定义一下就好了。

如果在 CSS 里遇到定义重复、冲突的情况，则后面的定义覆盖前面的定义。比如：

```
#miao {
    width:300px;
}
#miao {
    width:400px;
}
```

那么最终 #miao 的宽度是 400px，因为后面的覆盖前面的。

在 Html 文件里，link 一个 CSS 文件也就相当于把这个 CSS 文件的内容完全插入在那个位置，然后依旧是后面的定义覆盖前面的。这也就不难理解如果我们同时引入两个 CSS 文件，那么后面的文件优先级要比前面的文件高。

进而的，如果我把 style 标签写在页尾（其实这个标签不止可以出现在页头，它可以出现在几乎任何位置，只是不推荐罢了），那么他的优先级肯定是蛮高的，因为它在所有 CSS 定义的最后了。

但这还不算什么，元素的内联样式（就是直接写在元素标签里的样式）的优先级高于前边所有。

哇，好高哦！但是还可以更高一些，CSS 属性值末尾加上 `!important` 的优先级高于上面所有，比如：

```
#miao {
    width:120px !imporant;
}
```

这种写法可以覆盖元素的内联样式，因为他的优先级太高，非十分必要请不要使用！

但是还有木有优先级更高的？有！就是浏览器的扩展，他们的优先级高也只是基于上面的理论，将 CSS 写到页尾或者写成内联等，但是更可怕的是他们权限也是很高的，高到可以直接修改页面元素，所以我们测试页面的时候要禁用所有扩展，以避免不必要的干扰。

我忽然说这个你应该已经猜到了结局：我们只要把最后一个 li 元素的样式覆盖一下就好。没问题，这个有两种方式，一个是给最后的 li 都加上一个相同的 class，然后定义一下这个 class，演示如下（我只写关键部分的代码）：

```
<ul id="nav-items">
    <li class="nav-item"><a href="#">首页</a></li>
    <li class="nav-item"><a href="#download">下载</a></li>
    <li class="nav-item"><a href="#feature">特点</a></li>
    <li class="nav-item last-item"><a href="#about">关于</a></li>
</ul>
<ul id="nav-items-r">
    <li class="nav-item"><a href="#signin">登录</a></li>
    <li class="nav-item last-item"><a href="#signup">注册</a></li>
</ul>
```

注意一下： `class="nav-item last-item"` 的写法，空格分隔两个 class，也就是这个元素同时具有两个 class ，同时享受他俩的定义，还真是幸福呢～

```
#nav .last-item {
    border:none;
}
```

这个做法很容易理解，当然了，要注意新加的这部分 CSS 的位置哦，想想应该谁覆盖谁？

但是我很懒，懒得没边，再去加一个 class 倒是不是大事，但是我以后追加导航项是不是还要做修改，特别注意把 `last-item` 这个 class 加给最后一个导航项才对？你说这是多麻烦啊～

所以我想 CSS 的选择器能不能直接表达最后一项？能，真的能！，来看我的写法（Html 先复原，这次不用动他们）：

```
#nav .nav-item:last-child {
    border:none;
}
```

用冒号连接的叫做伪类，就是本来没有这个类，但是我们利用一定的规则去选择这个元素。现在很明白了，最后一个不加边框啊，于是我们的效果达到了。



但是你没有疑问么？我们的选择器好像是说的 `#nav` 里面的 最后一个 `.nav-item` 啊，怎么有两个 li 都没有边框了？不应该只是最后一个“注册”自己没有边框吗？

`:last-child` 这个伪类的意思是“属于其父元素的最后一个子元素”，很拗口是吧，打个比方你就懂了：俄罗斯小学开家长会（中国不行，都是独生子女，俄罗斯兄弟姐妹多一点），一个家长带着他家几个孩子，如此这般的家庭站满了操场。校长在上面说一至三年级的学生（这个相当于选择器）每家里那个最小的（:last-child 是每家最小的那个），上台来。这时候就要符合上面两个条件了，首先要是一到三年级的学生，然后如果一家有五个符合这一条件，那就只让最小的那个上去。能理解吧，你说我爸爸他们哥仨，我们兄弟姐妹一十八个是一个爷爷的，我们十八个人是不是只出最小的那个？这跟爷爷没关系，只看父母（父辈元素），父母相同的孩子里出最小的一个。

所以 li（.nav-item）的父元素是 ul，ul 他们家里中的最后一个，那么现在两个 ul ，当然也就出来两个最后一个。

大家好好看看上面的比喻，搞清楚 `:last-child` 哦，下一节我们学习新的伪类～

> 本章学习卡片：卡片 **23**
>
> 本章代码下载：本章代码

第二十四章 导航条（九）

很多同学问，你看他们那谁家的小谁做的那个导航，鼠标一滑过还有变化，他们怎么做的？

那今天我们来讲一些更好玩的伪类，首先是 `:hover` ，这个讲的是某元素被鼠标滑过的时候的样式。当然如果不定义的话就是没有变化喽，就像我们前边写的那样，但是如果我定义一下呢？

```
#nav .nav-item a:hover {
    background: #DDD;
}
```

然后我们去看看鼠标划过的效果：



看到变化啦，嗯，图上没有鼠标指针，但也猜得出来，我的鼠标放在下载上他的背景色发生了变化。你看他的文字大小啊、颜色啊，什么什么的其他东东是不是都没有变化？所以 :hover 是基于元素以前的样式进行变化的，我们只要把变化的内容写上就好了，相同的部分不需要重复的去定义。

原来鼠标滑过发生变化的代码竟然这么简单，曾经我们还当做一个什么什么的特效来看的啊！那么我们就再来点稍微复杂哦，看我改写成如下的样式：

```
#nav .nav-item a:hover {
    height: 40px;
    padding-top: 5px;
    line-height: 40px;
    background: #DDD;
    border-bottom: 5px solid #999;
}
```

我一下修改了好多内容，看看真复杂！但是仔细分析一下也就明白了，高度改成了 40px，所以 line-height 也改成 40px，这是为了垂直居中所做的相应的修改，而刚才高度上减少的 10px 则通过顶部内补 5px，底部增加 5px 的边框补回来了。因为上下是对称着添加的，所以文字依然垂直居中（看起来似乎没有变化），然后内补和边框都算作元素的范围之内（快递箱以及里面的填充物都算作整个箱子的范围之内），所以也不会影响 a 本身已经设定好的点击范围。

我们看看划过效果：



file:///home/dms/Dropbox/WorkSpace/coffee/show-code/index.html#download

就是酱紫的效果了，怎么样？大家根据自己的理解修改上面的属性试试看有什么变化？

然后再说几个 链接元素常用的伪类：

> :active 当前鼠标点下链接的时候
>
> :hover 当鼠标悬浮在元素上方时
>
> :link 当链接还未被访问过时
>
> :visited 当链接被访问过之后

现在我们也很少用这么多，设置一两种就不少了，比如导航部分的链接，设置一个 :hover 和一个 :active 就差不多，这个几个的用法一样，大家可以自行进行各种惨无人道的变态实验，以便使自己确实掌握这些伪类。

至此我们对导航条的美化也算简单的告一段落，如果你愿意，可以开动脑筋看看还可以做些什么样的美化，虽然我们现在学的 CSS 属性不是很多，但是通过有机的结合以就可以做出很多养眼的效果，希望大家多动手练习一下哦～

那么是不是我们的导航到此就算做完了呢？不是的，其实还有一个小小的问题没有解决。我们把浏览器窗口调小看一下哦～



你说，哎呀，怎么都出去？其实不是都出去了，而是根本就不在里面。不信你把 #nav 的高度去掉试试看，他的高度就会变为 0 ，根本没有被里面的元素撑起来。为什么会这样呢？因为里面的这些元素都进行了浮动，但是 #nav 并没有浮动。

浮动的和没有浮动的元素仿佛是不在一个频道的两群人，总觉的他们相互之间对不上频率，那怎么对上频率呢？两个办法：第一、浮动到底，要是所有的元素都浮动那自然就在一个频率上了，但是想想都觉得超麻烦；第二、清除浮动，就是在浮动结束之后清除一下，表示我浮动完了，咱们还是好朋友哦，不许再闹别扭啦，乖～

所以我们用完浮动之后一定要记得清除浮动，清除浮动的方法有很多，但是我们现在先来讲一种，是最容易理解的哦。

首先 #logo 和后面两个 ul ，以及 ul 里面的 li 都已经进行了浮动，那么在这样的局部范围内也算是全部浮动，于是他们之间相安无事，问题出在这些浮动的元素和外部没有浮动的元素 #nav 之间，所以我们要在 #nav 的内部这些浮动元素之后加一个： `<div class="clear"></div>` ，

于是变成了：

```
<div id="nav">
    <div id="logo">
        <a href="#">代码能有多难？</a>
    </div>
    <ul id="nav-items">
        <li class="nav-item"><a href="#">首页</a></li>
        <li class="nav-item"><a href="#download">下载</a></li>
        <li class="nav-item"><a href="#feature">特点</a></li>
        <li class="nav-item"><a href="#about">关于</a></li>
    </ul>
    <ul id="nav-items-r">
        <li class="nav-item"><a href="#signin">登录</a></li>
        <li class="nav-item"><a href="#signup">注册</a></li>
    </ul>
    <div class="clear"></div>
</div>
```

然后在 CSS 里面写上：

```
.clear {
    clear: both;
}
```

这样就可以了，为什么要这么写呢，这样定义以后，下次我们需要清除浮动的话，只需要复制 `<div class="clear"></div>` 过去就可以了，定义一次，可以多处使用，也算是很方便的，当然这种办法会多出一个空白的 div，虽然不影响界面，但是强迫症会比较难受，额......这个我们以后再讲其他办法。（强迫症别杀我......）

好了，现在我们去掉 #nav 的高度看一下效果。





这样就不会出来了……只是浏览器窗口较小的时候好难看………我们下节课再解决……（不要追杀我呀！）

> 本章学习卡片：卡片 **24**
>
> 本章代码下载：本章代码

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com** （刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发： http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书： http://www.jianshu.com/users/d37eaf3de0ff
- 站酷： http://www.zcool.com.cn/u/12927742

第二十五章 导航条（十）

现在来说说这个浏览器宽度的问题，这可不是一个小问题哦，因为大家的显示器尺寸是不同的，所以浏览器窗口的大小相应的也必然无法保持一致，我们必须同时考虑到可能出现的各种情况才行。

那么当前我们遇到的问题是当浏览器宽度过小的时候导航的内容放不下，于是被排成了几行，很不好看。当然这个情况如果不去管他其实也没什么问题，因为我们后面要写的其他内容都很宽，足以撑着页面保证宽度足够当前的导航使用。但是有些同学希望做的更严谨一些，那么我们来试一试。

我们先来简单分析一下，我们做的是一个电脑页面，#nav 元素宽度和页面相同，一般情况都是没啥问题的，但是当浏览器宽度太小，就会被挤压，也就是 #nav 的宽度不能太小就好了。

我们能说不让读者调节浏览器尺寸么？显然不行，那么我们规定一下 #nav 的最小宽度吧～这个很简单，CSS 中给给 #nav 元素加一个属性 `min-width: 960px;` 就好。这是什么意思呢？如果是宽度 `width:960px;` 的话我们就理解了，这是固定宽度 960px。现在呢是 #nav 享有根据情况自动调节宽度的权利（就像以前一样），但是，当他的宽度小于 960px 时，不可以了，到 960px为止，你便只能固定这个宽度，不能再小了。这就是最小宽度的意义，大了你随意，但是不可以小于这个下限，否则强制设定为下限的值。我们再来看看效果。宽度足够的时候不用说了，我现在把浏览器宽度设定为 800px，然后再看：



底下出现了滚动条，因为要是自由宽度的话现在 #nav 应该是 800px 宽，但是这小于他的最小宽度 960px，于是就被强制为了 960px。一个很好玩的属性。

那么有最小就有最大， `max-width` 是最大宽度。与此同理的高度也可以通过添加前缀产生两个属性，最大高度（ `max-height` ）和最小高度（ `min-height` ）。因为用法相差无几在这里就不细说了。

然后我们开始研究一下新的问题，这个问题也许有很多朋友早就等不及了，大家写这个的时候有木有想过宽屏的朋友？这导航分左右，在宽屏上左边一堆，右边一堆，中间空着一大块，那感觉就好像是下图：



你看，我们解决完了尺寸太小的时候的情况，现在还要面临尺寸太大的时候的情况，但是这个情况好像不是设置最大尺寸可以解决的问题啊。因为我们还是蛮希望背景填满整个宽度的，只是内容不要这么分散就好了。

等等，我们说内容不要这么分散！那就把内容部分打个包好啦～，是不是？我去打个包哦，你看着：

```
<div id="nav">
    <div class="nav-content">
        <div id="logo">
            <a href="#">代码能有多难？</a>
        </div>
        <ul id="nav-items">
            <li class="nav-item"><a href="#">首页</a></li>
            <li class="nav-item"><a href="#download">下载</a></li>
            <li class="nav-item"><a href="#feature">特点</a></li>
            <li class="nav-item"><a href="#about">关于</a></li>
        </ul>
        <ul id="nav-items-r">
            <li class="nav-item"><a href="#signin">登录</a></li>
            <li class="nav-item"><a href="#signup">注册</a></li>
        </ul>
        <div class="clear"></div>
    </div>
</div>
```

看到了，我又加了一个 div（.nav-content），然后把以前的内容都放在里面。现在我们设定一下他的 CSS。

```
#nav .nav-content {
    width:960px;
}
```

这个挺好理解的，我就是把内容限定在这个里面了，给他一个固定的宽度，至于外面的 #nav 让他带着背景去适应窗口好了，我们来看效果：



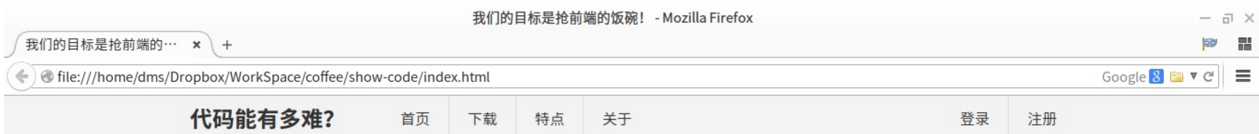
换做这么写倒是内容紧凑了，可是内容不在中间怎么看啊？来啊，这里教大家一个让元素横向居中的办法，啥是居中？就是元素两边空余的宽度相同，对吧？所以我们这样写：

```
#nav .nav-content {
    width:960px;
    margin:0 auto;
}
```

就是添加了一个 `margin:0 auto;` 哦，上下外补是 0 没有啥问题，本来我们就不需要，而且不设置默认也是 0。现在说左右，auto 就是自动啊，就是根据情况自己看着办，你想两边权限一样，当 #nav 的宽度比 .nav-content 大的时候，反正不是这边空着就是那边空着，两人权利一样大的结果就是——平分。这是最公平的办法，那么空余的部分左右平分是不是就是左右间距一样，就是居中了呢？

但是用这个要注意两个问题：第一、这个办法没法搞定垂直居中，因为受到文档流向上的原动力影响（参见 VIP章节 关于浮动的故事），同理浮动元素也不行；第二、被居中的元素必须指定宽度，否则两边都搞不清一共剩下多少，怎么平分？

来看看效果：



一切正常了，那么我们想一想，现在有里面的 .nav-content （已经指定了宽度）来撑着，#nav 的 min-width 还有必要吗？当然没有了，于是去掉它。

到这里，今天我们其实讲了导航的两种布局方式，一种占用全部宽度，一种只占用中心一部分宽度。希望大家好好练习一下。其实这两种方式都蛮常见，大家可以留意观察一下我们见到的网站哦。

然后我们发现我们花费了整整十章才写了一个简陋的导航，但是我们这十章只是在讲导航么？我们快把常用的 CSS 知识说的差不多了好不好？所以一定要认真照着做练习，多思考，觉得懂了就按着自己的想法去修改去尝试，这样才能让自己真的理解他们。当然要是不很懂也没关系，页面刚写到这里，后面的内容还要反复用到这些知识，一回生二回熟，跟着学下去慢慢也就会了哦～～

> 本章学习卡片：卡片 **25**
>
> 本章代码下载：本章代码

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com**（刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发： http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书： http://www.jianshu.com/users/d37eaf3de0ff
- 站酷： http://www.zcool.com.cn/u/12927742

第二十六章 CSS 书写规范

开一瓶一块八的啤酒，再来一串八毛钱的小鱿鱼，老板记得多放辣椒（这样经吃，这是经验，都记着点）。我们继续开始聊~我一个礼拜才三块钱零花钱，今天点了两块六的大餐，喵的，这日子不过了！

你说我今天废话特别多？是啊，这节课就是一节发牢骚的课。我们来解释一下题目哦，CSS是啥我们前边都说过了，书写呢？我们前面也都写过了，但是我们采用的是追加方式，就是需要什么就在后面加上他。规范，这个是我们今天要重点去解释的东西。

啥是规范？规范是没有强制性的东西，他建议你去遵守，遵守的话可能会给你带来一定的好处，但是你坚持不遵守也未必会有什么严重的后果。中小学生行为规范要求我们不要早恋，因为这会分散我们的精力，进而影响我们的学习成绩。但是你要是觉得学习这件事情简直是小菜一碟，轻轻松松就名列前茅了，或者你是富二代，别害羞，我们不会歧视你的，反正富二代其实完全可以不在乎分数什么的。那你完全可以选择这个美妙的人生副本（别吵，在学校阶段这个就是副本，毕业之后才是主线任务的）。这有问题么？规范有错么？我的解读有错么？

但是不止一次的有人跟我说，你这个不符合规范！你就是在误人子弟。可是为什么我同学里一对对早恋的都成了，而且很幸福。而且我的高中（某地一中，重点）的副校长的老婆也是高中谈的。什么是对的？合适的就是对的。非得拿着条条框框去抠的是教条主义，我写的又不是教科书，教科书又不是没有错。

你如果搜索《CSS 书写规范》网上有一篇流传很广，但是人家的标题是《推荐大家使用的CSS书写规范、顺序》看见了？是推荐，不是强制，不是必须，你又凭什么依据这个说我写错了？而且此篇文章的某些细节也不是大家都完全认同的，有兴趣可以去查查。因此而嘲笑他人写的如何的，我只能说：呵呵！

今天我说的规范是你不得不去遵守的，或者说是作为一个写代码的，起码要遵守的吧。当然其实我对你们没有任何要求，还是那句话，初学的时候达到效果是唯一目的，后面用的时间长了自然知道自己该怎么去做的。

原则是：第一、书写整齐，要么都缩进，要么都不缩进，缩进要整齐，总之就是好看。你想这么多代码，说看着不头疼是扯淡，如此情况下你还把他写成乱麻，反正你自己还要回头看的，你自己懂得。这个不只是 CSS ，写任何代码都是如此，这是为日后准备的，叫做有备无患。

第二、感觉将来可能不清楚的地方要加注释，这里还插一句 Html 的问题，最好每个 `</div>` 后面都加上注释，注明这是结束的哪个 div，否则嵌套多了后面查找的时候很费劲的。为啥你们的代码自己都懒得回头看，首先你的格式难看，然后就是没注释，自己再看的时候都晕了。

第三、CSS 元素的先后顺序，先外后内，先全局，后局部，所以 body 的定义在前面，因为他是最外面的，a {……} 这种也在前面，因为定义的是全局的，所有链接啊。然后按着从上到下，由外往里的顺序逐个写。元素的伪类既然是基于元素本来的样式的（比如 :hover）那他肯定跟随在元素本身的定义之后，元素都没定义，你基于谁去啊。

第四、内部的属性按着从整体到局部的顺序去写，搞不懂就自己大概有个顺序，各个元素都按照这个顺序就好，因为以后查看的时候方便点，要不然几个元素都是十几个属性的，然后一个宽度写在最上边，一个在最下边，神仙都气哭了，完全不知道你下一张牌怎么出啊。

第五、注意点有前因后果的，把基础条件写在前边。本身是内联元素，你不先 `display:block` 又哪来的 `float:left;` 啊。我们说 `margin:0 auto` 居中的话必须有固定的宽度，那你是不是该把宽度定义在他前面？这就是个逻辑关系。

然后少点重复定义啥的，挺好了。还有 id 和 class 尽量命名的让人一看就懂这是干嘛的，反正你故意反着写，用 #footer 当页头也没人管你，让下一个看你代码的人去哭吧，在特殊情况下又不是没人这么做，如果你确信你不会再看这个代码完全可以试试，但是要是挖了坑有自己跳进去就呵呵了。

就这样差不多够用了，写的多了你自己就给自己一份必须遵守的规范了，其实哪用得着我来废话。技术是让人来享受的便捷的，不要刻意地把它复杂化。别觉得学代码很可怕，其实编程这件事情也在努力变得更加友好，所以前面我们的各种不规范也都被浏览器容忍了。

那么大家试着去把我们已经写过的代码规范一下吧~我也简单修改了一下，大家可以下载看一下。

没看懂的记得以后过来看~

> 本章学习卡片：卡片 **26**
>
> 本章代码下载：本章代码

第二十七章 首屏大海报（一）

一块八的啤酒真不经喝，没聊两句就喝光了，导致上一章有点短。现在还剩下四毛钱，买个棒棒糖都不够，话说有钱不知道怎么花也是一种折磨啊。

今天开始聊这个页面的大海报部分，但是一个关键的问题是：首屏。这是页面的灵魂所在啊，好在我不用考虑这些，设计的问题交给设计师去发愁好了，所以我今天就随便找张图片演示一下就好。

你真的这么去想？太好了，你又被我忽悠了，其实今天是要你们看看前端如何灭掉设计师的。别反驳我，看下去你就知道设计是死的，网页是活的了。

我们首先确定海报是宽屏的，就是屏幕多宽，我的海报就有多宽，这是为了让老板感觉到：“”哎妈呀！你们做得好震撼！“好的，我们开始添加 Html 代码，应该按着顺序加载 #nav 部分之后。

```
<div id="post">
</div><!-- #post End -->
```

注意，我现在开始更加规范的写代码了，所以注释一定要跟上，随着代码的增加，越来越容易让我们看迷糊了，所以注释这个小秘书显得愈发性感起来了。

我们知道 Html 最多只是骨感美，想要丰满，必须”创世神“（CSS）。

```
#post  {
    width: 100%;
    height: 450px;
}
```

现在设计师可以骂街了，你整天问的（别说没有，群里我被我问了可是不止一次两次的）网页我该设计成什么宽度的问题，在这里可一点麻烦都没有，一个 `width: 100%;` 明确解决了。你说那还没放图片呢，说这个为时过早吧？其实放上图片你哭得更惨。

我先去准备一张图片哦，要够大的才够爽，嘿嘿~于是我随手找了这张，仅作演示而已。挺大的，2048×1363 像素的，我懒，就不根据需求裁剪了，你们不许像我这么懒，这样不好，因为图片过大会导致加载过慢的。好了，先欣赏一下这张图片：

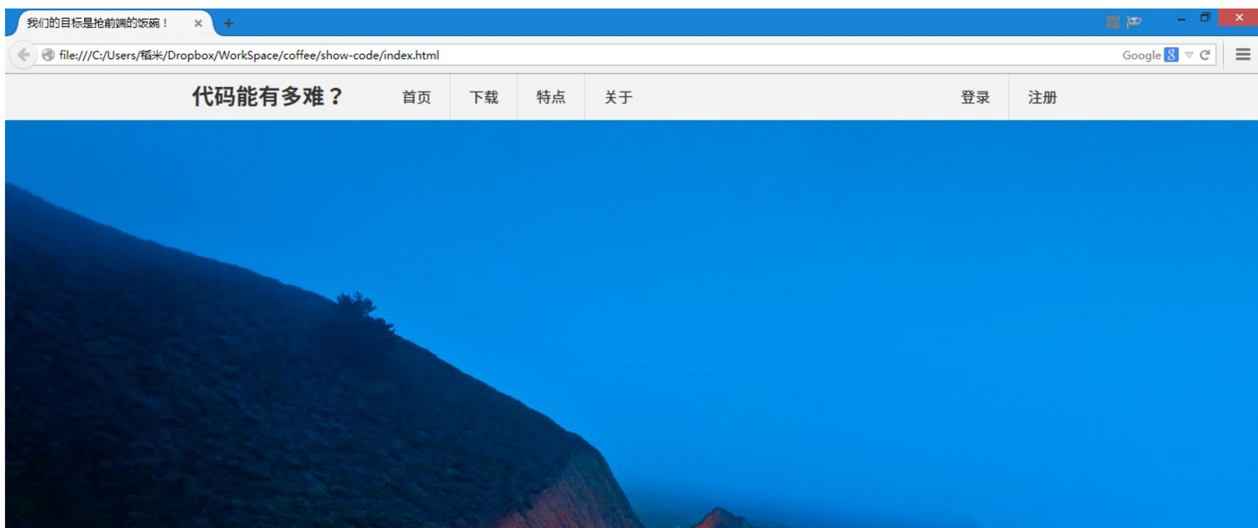


然后开始设置背景了哦，其实很简单的，以前我们设置纯色背景用的是：`background:颜色值;` 现在改成：`background:url(图片地址);` 就好了。这里说一下，`background` 是个复合属性，其实分开写的话有个七八十来个，我记不住，我就记住了一个，就用它了，我省心，它累不

累的又不关我的事......其实这样更简单明确啦，分着写......字多！

```
#post  {
    width: 100%;
    height: 450px;
    background: url(../images/bg.jpg);
}
```

简单吧，来看看效果哦~



什么，你的图片没有显示？右键——审查元素，你会在右侧 CSS 里看到图片的地址，然后点击或者右键（根据不同浏览器操作略有不同）打开图片，然后查看地址栏，看看图片地址跟他的实际地址有什么区别，然后据此修改代码。这可是一个很好的练习机会，希望不要错过哦。

不过其实你要是按着前边我们说的文件结构去做的话，这边也照抄基本不会有什么错，只是错过了一个练习而已。

好了好了，其实现在图片位置并不合适，我们看到整张图片就显示了一个左上角。我们要定义一下背景图片的位置（ `background-position` ），但是在这之前我们有必要先讲一下网页的坐标系。坐标系三要素，原点，坐标轴，正方向（恩恩，其实我说错了，应该是......你管呢，我现在就想说这三个，就这么地吧~）。原点，以参照物的左上角为原点，坐标轴有三个：X 轴（横向，即左右）、Y 轴（纵向，即上下），Z 轴（深度，即前后）。学过三维的同学说，咦，这个好熟悉的说。然后正方向，X 轴向右为正方向，Y 轴 向下为正方向，Z 轴 向屏幕外是正方向。

你没听懂？请打电话给你的初一几何老师，让他数落你一小时三十二分钟。基本上本教程用到的最深奥的数学知识也没超出初一初二的水平，而且也只是用到最基础的一点概念而已，公式什么的......不好意思，我也早就还给老师了（别打，老师我是真的从来没用到过他们啊！）

其实网页上涉及的坐标问题很复杂，但是我的原则是，记不住我干脆不记，用的时候简单测试一下就知道了。

背景位置的属性值分为两个，一个 X 轴，一个 Y 轴，他们都可以用数值去表示图片的起点（左上角）坐标。然后我们漏说了一个问题：原点，我们以本元素的右上角为原点。你说你隐约知道怎么调整背景位置了，Too 森普！ X 轴还有三个特殊值：left、center、right；Y 轴也还有三个特殊值：top、center、bottom。可以顾名思义吗？就是以图片的左（中、右、上、中、下）边与元素的左（中、右、上、中、下）边对齐。很复杂？看懂了就行，看懂了用的时候自然就会想起来了。

这就完了？还有百分比呢，喵的，w3schools 的网站不管是中文和英文都没仔细去解释这个问题，百分比和数值是有根本的区别的，他跟那几个特殊值倒是很像，这里只简单的说一下：

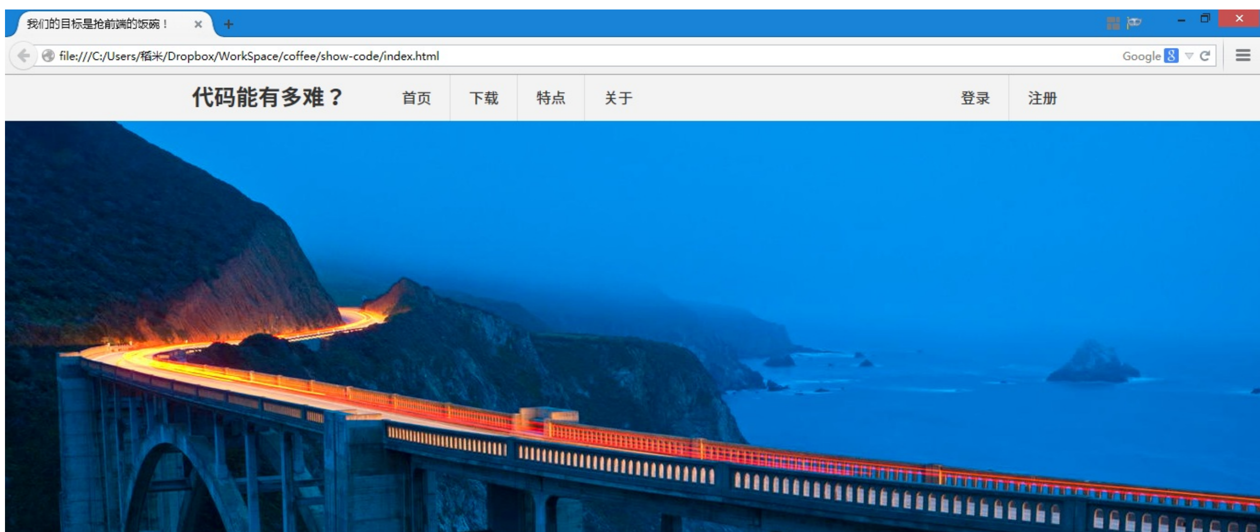
> `background-position: 0% 0%` 相当于 `background-position: top left`
>
> `background-position: 50% 50%` 相当于 `background-position: center center`
>
> `background-position: 100% 100%` 相当于 `background-position: right bottom`

其余数值自己体会。然后再说一下，w3school 说："提示：您需要把 background-attachment 属性设置为 "fixed"，才能保证该属性在 Firefox 和 Opera 中正常工作。"这句话在英文版 w3schools 中并未提及。然后中文的 w3school 只是对英文 w3schools 第三方翻译，而不是官方出品。然后其实浏览器发展的那么快，现在根本不需要这个设置了。所以，奉劝一句，那些把 w3schools 奉为圭臬的同学醒醒吧，拿他当词典也就好了，但词典不是标准，只是参考。而且如果可能，尽量直接看英文版。好吧，我又多事了，管这干啥......你爱信谁信谁，反正我说的又不全对。

回到正题，现在我们的背景应该如何设置？横向居中对齐，用 center ，纵向要往上一些，以便保证最美好的位置得以展现，这里用固定值就好，所以用像素来表示，那么图片在 Y 轴的起点向上偏移，上面是负方向，所以用负值表示，于是我们写作 `background-position:center -250px;` ，不过我懒，于是把它加载了 background 这个复合属性里：

```
#post  {
    width: 100%;
    height: 450px;
    background: url(../images/bg.jpg) center -250px;
}
```

来看看效果哦~



我觉得单这一个属性就很好玩了，大家可以试着去做各种设置，看他的变化，然后想象一下在什么情况下需要这种设置。

> 本章学习卡片：卡片 **27**
>
> 本章代码下载：本章代码

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com** （刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发： http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书： http://www.jianshu.com/users/d37eaf3de0ff
- 站酷： http://www.zcool.com.cn/u/12927742

第二十八章 首屏大海报（二）

据说看完上一节教程之后好几拨人哭昏在了厕所。别再昏了，起来嗨~

其实后面讲到响应式我们可以根据实际情况去调节海报的高度，讲到 JQuery 我们可以很方便的动态设置海报高度。那时候有的你们哭的，让感动的泪水尽情地挥洒吧，欧耶~

反正也都感动了，索性再感动一下好了，来玩个 CSS3 的属性哦~对了，你们说什么是 CSS3 啊，什么是 HTML5 啊，就是那些低版本 IE 兼容不了的东西……

所以今天的属性你要是在小于等于 8 版本的 IE 下浏览可能不起作用。这个属性是 `background-size` ，他可以同时含有两个值，依次是 X 轴方向和 Y 轴方向。可以设置带长度单位的数字值，或者百分比。然后不设置则默认 auto。这个跟 `width` 或者 `height` 属性是一致的，其实这就是设置背景图片的宽高，你看这个以前让很多人发愁过的问题到了 CSS3 下面变得十分简单了。

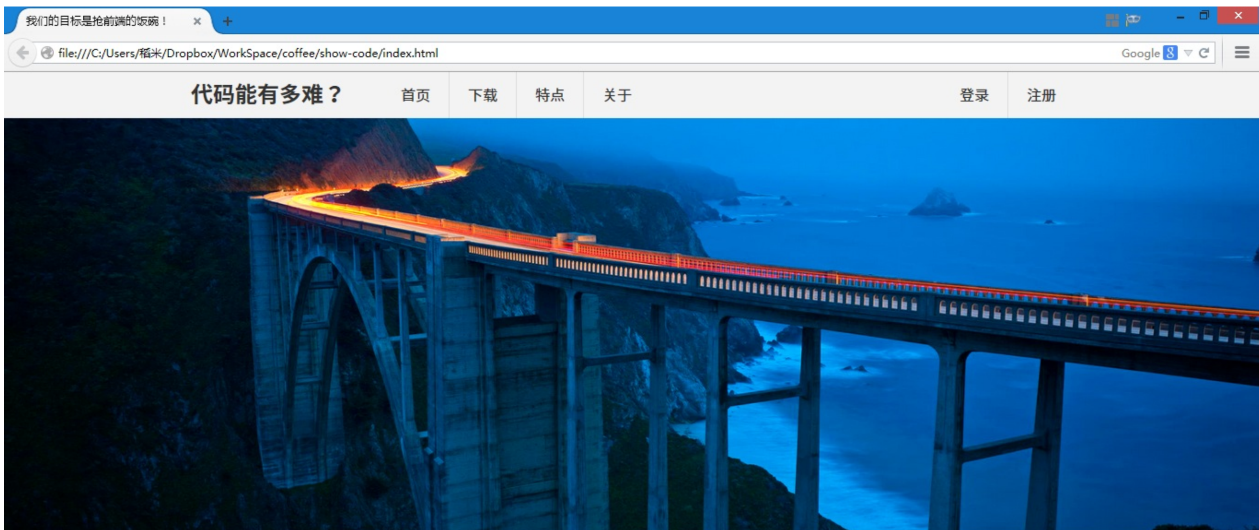
只是宽高设置还需要演示么？配合昨天讲到的（ `background-position` ）可以想出各种玩法，看你的想象力了哦。然后我们再来说一个 `background-size` 的特殊值： `cover` 。这个值什么意思呢？就是图片等比例缩放至恰好覆盖整个元素，这种情况其实说白了就是某一边正好 100%，另一边等于或者大于元素对应方向的长度。

反正你理解为图片不变形（只是放大或者缩小）的填满元素就是了，再不理解就试试呗，反正不要钱。这就是学习方法，你看，这么说就显得高大上了。

然后我们这里就用 cover 这个值就好了。

```
background-size: cover;
```

看看效果，大图在横向上得以完美地展示：



但是这里存在一些隐含的问题，第一、我们当前图片较大，需要进行偏移（background-position），那么我们的偏移量就很难确定，不过如果你只是定位（left、top 之类）倒是还不至于有影响；第二、我们的海报（#post）长宽比不是确定的，所以，你可能无法确定究竟哪一边是 100%，进而的，你也可能无法十分明确的确定海报的展示区域。在当前，这是我们很难解决的问题，所以这次我们无法使用这个属性。但是等到后面我们学习了响应式和 JQ 之后，可以根据情况对网页进行动态设置了，那这些鸡肋的问题就可以解决掉了。那才是真正有了生命的网页。

好啦，演示了一个 CSS3 属性，其实也不过如此嘛，没有什么神奇的地方，跟以前的属性用起来一样的普通啊，而且他也有他的小缺点。可是有些人非得神化这些东西，把 HTML5 和 CSS3 搞得多么多么高大上，其实完全没必要么，反正我就是个土鳖，给我披萨我也卷着大葱吃，但是我身体倍儿棒，你能怎么样。

反正也是讲到背景图像嘛，索性我们就再讲一个属性好了。 `background-attachment` ，这个属性是说元素的背景图像是否相对于窗口滚动，默认当然是滚动了，就是像我们平时见到的那样。但是当你设置为 fixed 的时候，他就相对于窗口固定了，测试方法：给咱们现在的代码的 #post 元素加上 `background-attachment: fixed;` ，然后在 html 代码里的 #post 元素下面加上很多个 `<br />` 这样页面就可以滚动了哦，那么上下滚动页面试试看吧，随便你的页面怎么动，背景也在原地的哦。借助这个属性我们又可以做出很多神奇的效果了呢。当然，这里也说明了上一章提到的 w3cschool 的说法是怎样的误导了。

然后，我们把 `background-attachment: fixed;` 去掉哦，那些多加的 br 自然也要去掉，回到我们的主线任务上来。我们的主线任务是什么来的？哦，对了，吃烤串，老板，再来三十串大鱿鱼，少放辣椒，对，就是对面这个正在看我教程的人付款。别跑，还有啤酒呢……算了，他跑了，我们继续聊海报。

这是一个海报，假设我们现在我们的海报上已经有了完整的内容，你别看我，我这上没有，我就放了张图片示意一下，看不惯你咬我啊，额，男生就算了。那么我们现在想给他加个链接，这是一个很常见的需求，对吧。你说要是图片就简单了，可是我们却把它设置成了背景。我跟你说啊，这个要是图片，你想让他这么居中对齐，然后两边超出的部分隐藏，这个效果可是要好麻烦才能做出来的哦。而且你真的打算给宽屏图片整个加链接吗？那你一定没考虑过用户体验这一层。

我在一个页面需要滚动的时候喜欢先用鼠标在空白的地方点一下，以确保焦点在页面上。但是如果我点哪里都会产生某些作用的话，我会很困扰。就好像你要抓住一块掉落的豆腐，却又不能让他碎掉般纠结。所以我觉得我们把两边只是为了视觉补白的部分留出来，只给中间有效内容区加链接就好。

怎么加？其实完全是我们讲过的东西哦，先给 #post 元素里边放一个空链接，我说的空是指链接内没有内容，不是说不设置链接的各种属性哦。

```
<div id="post">
    <a href="http://www.google.com" class="post-link"></a>
</div><!-- #post End -->
```

然后给刚才的链接设置 CSS。

```
#post .post-link {
    display: block;
    width: 960px;
    height: 100%;
    margin: 0 auto;
}
```

a 是内联元素（inline），先转换成块元素，然后设置宽高，最后用 margin 居中，很简单。但是我们的目的达到了，去页面移动鼠标看看吧，注意鼠标光标的变化哦~~

然后说一下这个宽度，960px 是在 1024px 宽度的屏幕下可以舒服的展示（不出现横向滚动条）的最大宽度。有人说淘宝的 990px 宽度，不过不幸的是掏宝商城页面在 1024px 宽度屏幕下会出现横向滚动。也许你说这不重要吧，反正现在都是大屏幕，这就看你期望怎样的兼容性了。不过其实有了响应式，这些问题也就不是大问题了。可是到了低版本的 IE 上，唉，一切都是浮云，不得不再说一句 IE 该死！

> 本章学习卡片：卡片 **28**
>
> 本章代码下载：本章代码

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com** （刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发： http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书： http://www.jianshu.com/users/d37eaf3de0ff
- 站酷： http://www.zcool.com.cn/u/12927742

第二十九章 首屏大海报（三）

不过显然的，要是做个海报这么简单码农们就不哭了。所以现在我们要做一些没有设计师我们一样玩的很 High 的工作。

你看我的海报其实还只有一个背景，上节课虽然加了链接，也不过是自欺欺人，没内容谁点击啊？所以我们要开始添加内容了。

那么上一节加的那个链接就显得不够精致，所以不要他了，于是我们好像又回到了第二十七章结束的时候。注意哦，现在每个元素我都在讲着他的多种可能性，你们一定要注意点，跳着看的话可就乱了哦。

首先做什么？首先要给定内容一个范围，这样比较便于控制里面的内容，否则我们基于宽度变来变去的 #post 进行定位就会很困扰，要考虑的可能性太多了。但是如果我在其中划出一片固定大小的区域用来放内容，事情就变得简单了。

划定一个区域，肯定是居中的，能够被各种屏幕正常显示的区域了，然后高度就是 #post 的高度，等等，怎么感觉越说越像昨天加的链接了？

```
<div id="post">
    <div class="post-content">

    </div><!-- .post-content  End-->
</div><!-- #post End -->
```

```
#post .post-content {
    width: 960px;
    height: 100%;
    margin: 0 auto;
}
```

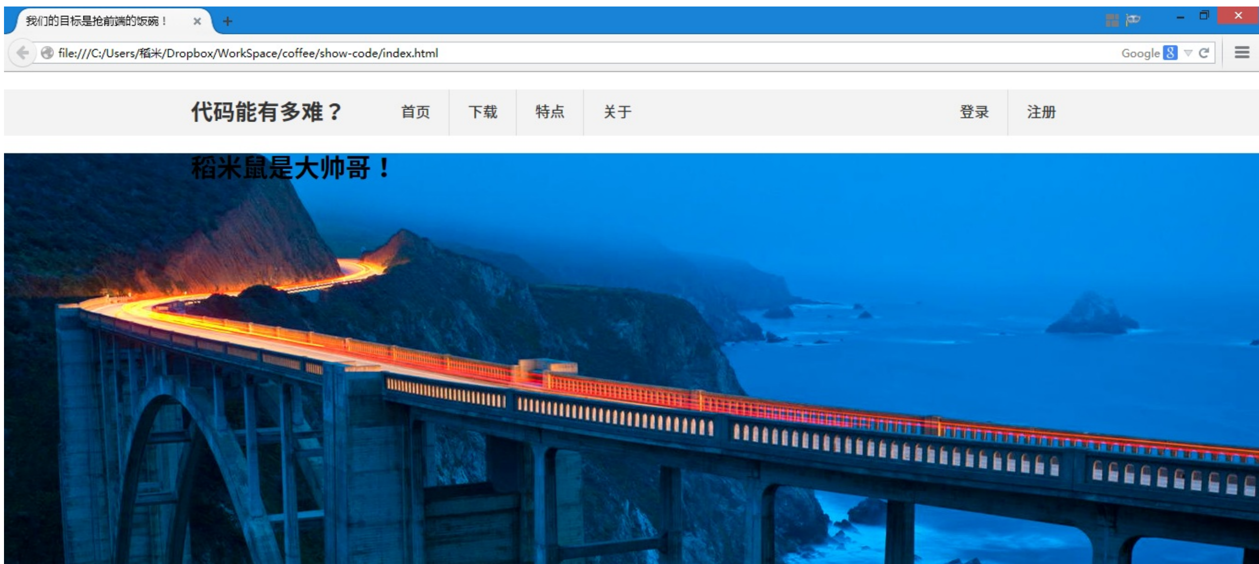
相似度确实不低呢，所以啦，其实写网页就是那点东西，套来套去的，套的多了自然就熟练了，也就会用了。

在这之后是什么？往里面加东西啊。来，先加上这句：

```
<h1>稻米鼠是大帅哥！</h1>
```

当然了，我们知道这是一个不容置疑的事实。但是，你应该知道吧，上边这句要放在 .post-content 里面哦，别放错位置。

然后如果你刷新页面的话……你会从这一刻开始怀疑人生。



先说 h1 标签是什么，是标题，标题标签有六个 h1—h6，依次从大到小。这么说这个标签肯定又是有自带样式的了，当然我们可以根据需求欢快的去重新定义他们。那么现在的问题是怎么引起的呢？h1 自带了 margin 属性，但是 margin 可不单纯，这个属性用起来很复杂，所以我特别建议一下，如果可以用 padding 的情况最好不要去用 margin ，除非你真的足够熟悉它的特性。

那么现在我们怎么做呢，我先打一个不太恰当的比方去理解一下当前的情况，padding 是内补，内部问题终究是好解决的。margin 是外补，这是一个要跟他的相邻元素或者父元素产生相互影响的事情，于是就有点热闹了。眼下的问题是父元素对他的约束力不够，就是当爹的没管住孩子，孩子就开始淘气了。怎么管住孩子呢？加一些特别的属性就可以的，这个可以根据情况选择各种属性，比如给父元素加个内补神马的，但是这里我们并不希望加内补，因为内补会影响元素的尺寸。那就加一个语义上说得过去的属性好了，反正我也挺随心的。

那么给父元素（#post .post-content ）加上一个 `overflow: hidden;` 好了，你说这个什么意思？这倒是很值得说道一下。overflow 这个属性是说元素内容如果超出了元素的范围怎么办。这就比方说你一个高二百的元素里插入了一个高五百的图片，多出来那三百高度怎么办？我们后面的属性值也给出了答案，hidden 隐藏。藏起来不让人看到就是了哦。

然后刷新页面，你看到终于算是符合预期了。现在就开始下一个问题，标题居中。这个很简单了，文字水平居中啊，居左啊，居右啊，用这个属性来定义： `text-align` 可用的属性值有 left、center、right 等，这个比较显而易见，不多说，顺便把文字颜色（color），距离顶部的距离（margin-top）定义一下好了。

```
#post .post-content h1 {
    margin-top: 120px;
    color: #EFEFEF;
    text-align: center;
    font-size: 42px;
}
```

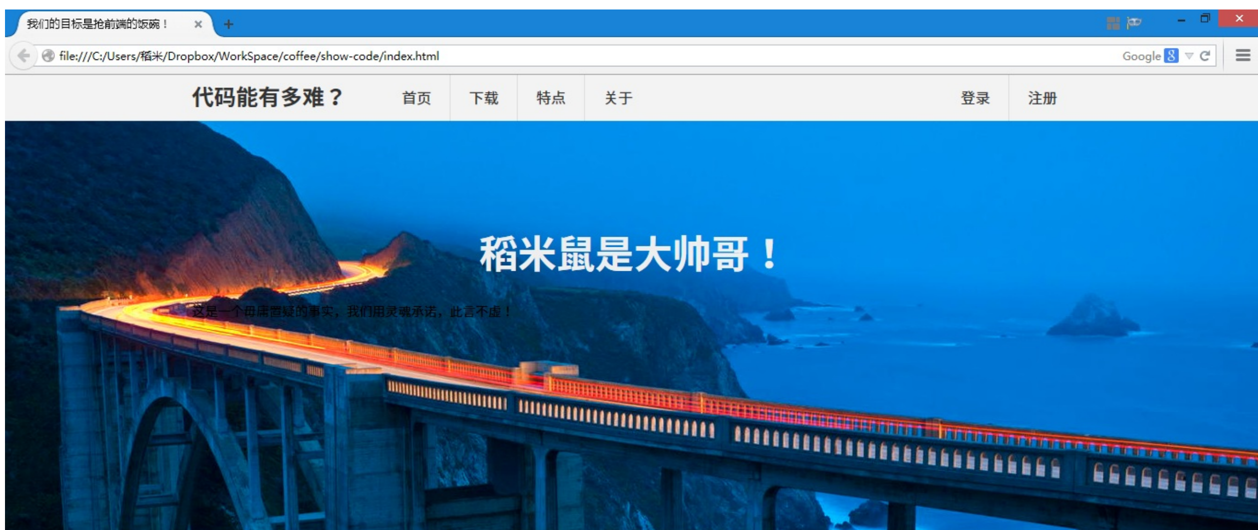
看看效果：



嗯，为了强调这个事实，我又把字号（font-size）改大了一些，这样感觉更有说服力啊。但是我怕你们还是无法正确理解我的帅啊~所以我还要在这个标题底下加上一行小字说明一下。当然就是在 h1 元素下面加上：

```
<p>这是一个毋庸置疑的事实，我们用灵魂承诺，此言不虚！</p>
```

然后看效果：



说真的，我仔细找了半天才找到那一行小字。现在我们很容易想到给他加上各种样式，不过仔细想想，我们要加的样式跟上面的 h1 元素好像有很多重复啊。这时候其实我们没必要定义两次。那个同学说用一样的 class ，没错！这只是其中的一个方法，现在我们来试试另外一种。我去把 h1 的文字颜色（color）、文字对齐方式（text-align）的定义，移动到 `#post .post-content` 元素的定义里。变成了如下的样子：

```
#post .post-content {
    width: 960px;
    height: 100%;
    margin: 0 auto;
    overflow: hidden;
    color: #EFEFEF;
    text-align: center;
}
#post .post-content h1 {
    margin-top: 120px;
    font-size: 42px;
}
```

来看效果哦:



你看实现了，但是我并没有去定义更多的内容，或者说没多打几个字。这是 CSS 样式的继承，即子元素会继承父元素的一些属性。比如文字相关的属性，大部分都是可继承的。

> 本章学习卡片：卡片 **29**
>
> 本章代码下载：本章代码

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com** （刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发： http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书： http://www.jianshu.com/users/d37eaf3de0ff
- 站酷： http://www.zcool.com.cn/u/12927742

第三十章 首屏大海报（四）

> 如果本章演示效果无法正常展示，请至主站查看 http://coffee.zji.me/

你看，代码这东西认真玩起来其实还是很有意思的。不过可能有同学问了，咱不是写一个 App 的发布页面么？怎么……啊啊，是啊是啊，所以我们接下来要写一个下载按钮呢~

这个按钮的写法比较多啊，你要是只为了链接到一个文件，其实用 a 标签就可以搞定啊，你会设置背景吧，会设置文字颜色吧，会设置边框吧。然后哩？

当然，要是希望用来发生什么有趣的事情，或者为了语义明确，我们也可以选用 button（按钮）标签。这个写起来超简单的~

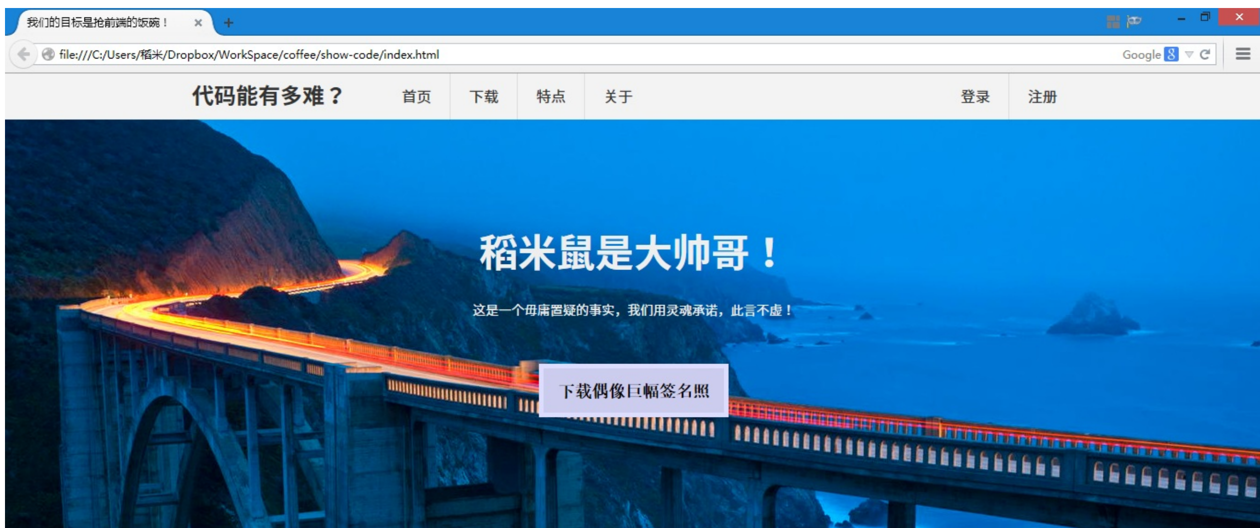
```
<button>下载偶像巨幅签名照</button>
```



是挺简单的，也挺难看的，开始给他定义样式：

```
#post .post-content button {
    padding: 12px;
    margin-top: 36px;
    background: #CCE;
    border: 5px solid #DDF;
    font-size: 18px;
    font-weight: 600;
}
```

大家应该都看得懂的，我也不多说了，看看效果吧，请原谅我这诡异的配色！



到这里，这个海报其实就做得差不多了。虽然很多细节没有去完善。不过......那不是应该你们自己去做的吗？赶紧的，每个人做一个漂亮的出来。

本来这章已经可以结束了，但是好像字数太少了，我总要对得起各位读者的票钱......虽然你们大部分没给钱。所以在下面做一些简单的补遗。

第一个问题：背景相关的还有一个属性：background-repeat 是说背景重复。默认是重复的呦，就是背景图片小于元素的话，则通过重复的方式进行填满，而我们也可以单独设置其中某个轴是否重复。

然后我来举个例子：`background-repeat: no-repeat repeat-y` 就是横向不重复，只在纵向重复。挺好理解的事情，就不多解释了。

定义背景的时候我们可以把这些属性都写在一起，为什么这么做呢？我就是图省事......

```
background:url(miao.jpg) center -15px no-repeat;
```

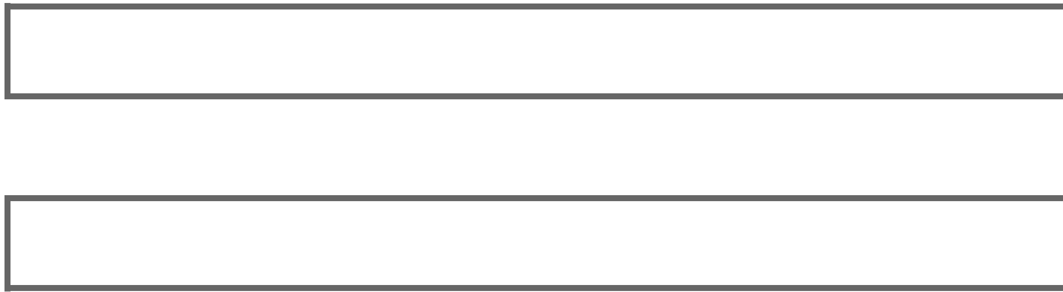
然后说说 margin 这个家伙。这个家伙其实也没有想象的那么麻烦，就是纵向上两个 margin 相遇会重叠。比如：

```
<div style="height:50px;border:3px solid #666;margin:50px;">

</div>
<div style="height:50px;border:3px solid #666;margin:50px;">

</div>
```

效果如下：



你看你看，两个元素的间隔明显不是 100px（50+50）啊。其实只有 50px ，因为两个 margin 重叠了。

然后其实上一章 h1 引起的问题也是着个问题的一个变种，当时我们已经解决掉了。这个先知道就好，剩下的到实践中去理解。

然后留个作业吧，敢不敢独立把这个页面完成掉？而且要尽可能的完美~

> 本章学习卡片：卡片 **30**
>
> 本章代码下载：本章代码

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com**（刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发： http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书： http://www.jianshu.com/users/d37eaf3de0ff
- 站酷： http://www.zcool.com.cn/u/12927742

第三十一章 还是海报

报告读者，本人正在直播......直播写作，接受监督，但是我就不告诉你们在哪里直播。

以下为注释 `<!-- http://live.bilibili.com/21024 -->`

好了，今天废话不多说，开始我们的正文。你的啤酒准备好了么？

前边我们写了导航栏、写了大海报，你也许觉得这还是在入门，好简陋好简陋的，但是其实呢，这已经是我们常用知识中的大部分了。不信你看，今天的内容几乎完全是在复习的样子啊。

那么呢，作为一个 App 发布页面......额，虽然已经被我写得不伦不类的，你就当成那个样子去看好了，毕竟我这么帅，你们会原谅我的对不对？

好啦，为了展示它的特性呢，它是谁？不是你想的那个样子啦，是为了展示我们要发布的这个根本不存在的 App 的某些引人入胜的特性，我们要多加几张海报。你说把前边海报部分复制一下就 OK 啦，反正也是，但是我要是那么懒，你们还不打死我？再说了，千篇一律的版式也不方便糊弄老板啊！所以下边两个海报的样式总要变一变的。

第一张呢，左边文字，右边一张什么什么 LOGO 或者 ICO 什么的，哎呀，不说那么高大上了，放个剪切画的意思。第二张就是把这两边换换，左边剪切画，右边文字，当然啦，他们都有通栏的背景，这样显得符合潮流......其实是符合老板恶俗的审美啊。

那么开始我们的工作，第一张海报......算了，这么懒的我肯定是两张一起写了，首先是 Html：

```html
<div id="post-a">
    <div class="post-content">

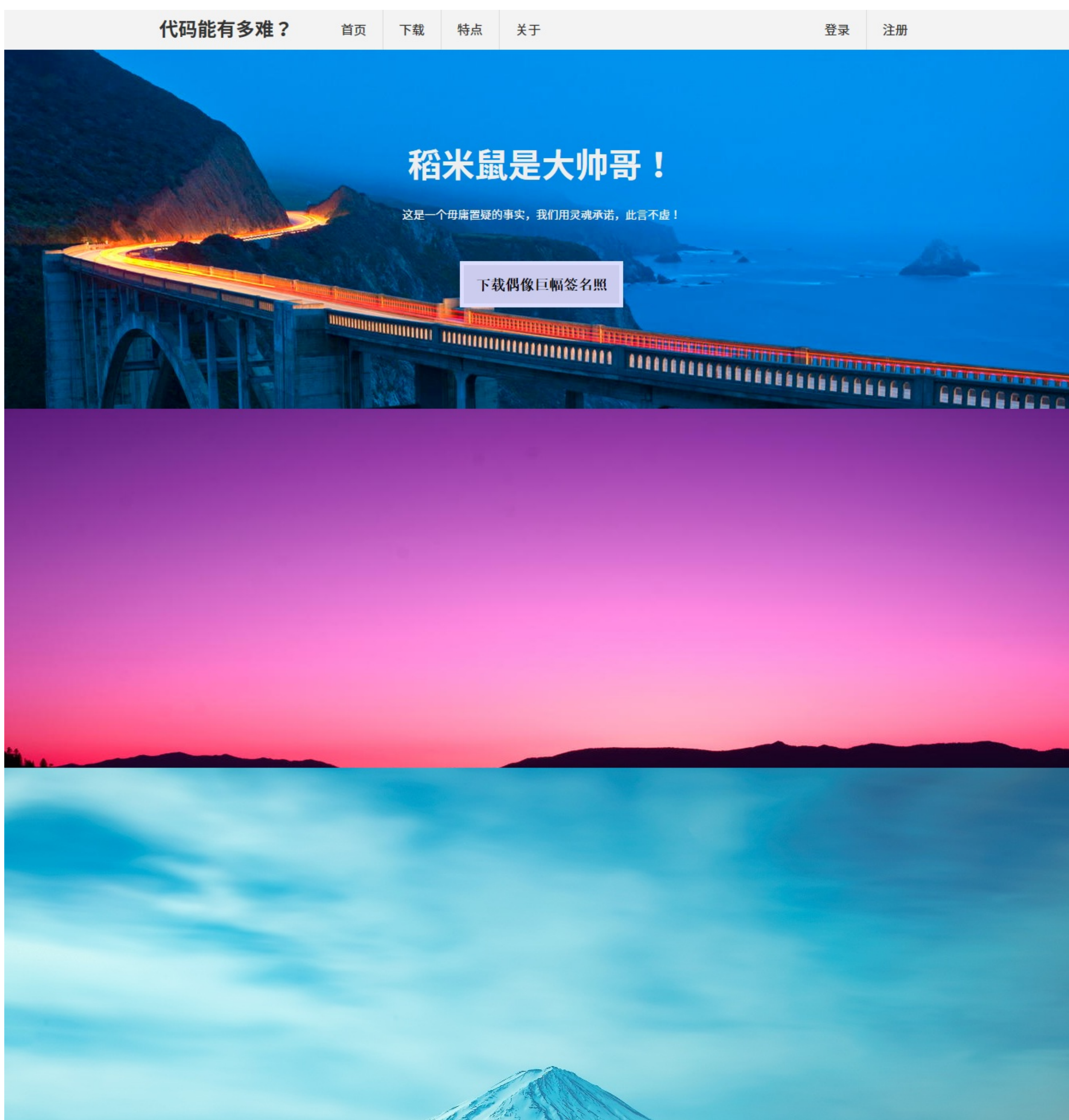
    </div><!-- .post-content End -->
</div><!-- #post-a End -->
<div id="post-b">
    <div class="post-content">

    </div><!-- .post-content End -->
</div><!-- #post-b End -->
```

然后是一些基础的 CSS 样式控制，我不用讲你也看得懂啦。你看你们老说学不会啊、看不懂啊，现在为啥就能看懂了？切，根本就没那么难嘛~

```
#post, #post-a, #post-b  {
    width: 100%;
    height: 450px;
}
#post {
    background: url(../images/bg.jpg) center -250px;
}
#post-a {
    background: url(../images/bg-a.jpg) center 0px;
}
#post-b {
    background: url(../images/bg-b.jpg) center -250px;
}
.post-content {
    width: 960px;
    height: 100%;
    margin: 0 auto;
    overflow: hidden;
    color: #EFEFEF;
    text-align: center;
}
```

以上是我修改/添加的 CSS ，大家看看发生了什么变化？为什么捏？#post, #post-a, #post-b 三者样式区别不大，所以干脆放在一起定义一下相同部分，然后再分别设置背景好了。.post-content 样式是可以共用的，那就共用好了，所以去掉 #post 的约束。然后来看看效果，这次有点长……

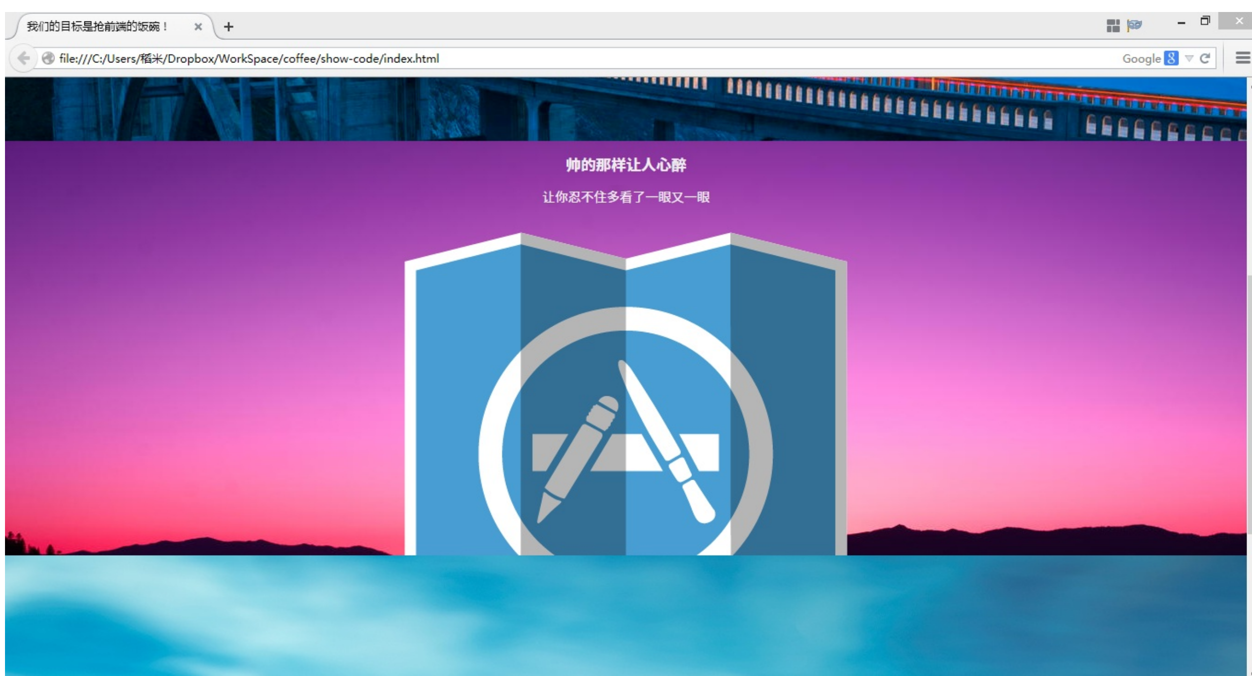


现在开始填写海报里的内容，几乎跟第一张海报是一样的。什么什么？你说我偷懒？分明就是复制了第一张海报在糊弄？其实你把下两张海报的高度改低点（或者高点），然后设置个纯色背景就挺高大上的，不过好像比现在的还简单……所以我没偷懒好伐？！

现在给第一张海报加上一个图标，一段文字。Html 代码如下：

```
<div id="post-a">
    <div class="post-content">
        <div class="post-text">
            <h3>帅的那样让人心醉</h3>
            <p>让你忍不住多看了一眼又一眼</p>
        </div>
        <div class="post-icon">
            <img src="images/icon-a.png" alt="">
        </div>
    </div><!-- .post-content End -->
</div><!-- #post-a End -->
```

我多复制了两层，所以呢，你应该可以知道这代码是写在哪里的了。首先我们去看看当前混乱的局面，然后我们再去规范他的样式。



看看，随手一放，就是这么的别有风味，现在我们开始设置一下他的样式，等设置完成就变得俗不可耐，十分讨老板欢心了。

```
.post-content .post-text {
    float: left;
    width: 510px;
    padding-top: 165px;
    font-size: 28px;
    line-height:  60px;
}
.post-content .post-text h3 {
    margin: 0;
    font-size: 36px;
}
.post-content .post-icon {
    float: left;
    width: 256px;
    height: 256px;
    padding: 97px;
}
.post-content .post-icon img {
    width: 256px;
    height: 256px;
}
```

这些 CSS 我就不逐一的解释了啊，你们可以通过审核元素去理解他们为啥那么写。两个元素都 `float:left` 是为了横排；图片占了 450×450 的面积，左边剩下 510宽的面积给文字，通过一些属性的设置使得文字在他的位置内居中。

然后效果如下：



这基本就达到了我们的预期，当然要说严谨的话还要清除一下浮动才好，当然如果不清除其实也出不了什么问题，因为各种属性的限制，现在的布局其实不至于乱掉，当然只是表面上看起来，事实上道理并不是讲的很通，反正加不加看你喽。

现在第二张海报......其实好简单的，只要你把 Html 写一下就好。

```
<div id="post-b">
    <div class="post-content">
        <div class="post-icon">
            <img src="images/icon-b.png" alt="">
        </div>
        <div class="post-text">
            <h3>其实做人要谦虚的</h3>
            <p>你看作者从来不说自己帅</p>
        </div>
    </div><!-- .post-content End -->
</div><!-- #post-b End -->
```

然后就行了呢，因为他要用到的 CSS 我们在前边都已经定义了呢~好啦，看一下整体的效果就可以结束这节课了哦。（其实底下还有悄悄话，我不告诉你）



你看，你看，这效果还是看得过去的......吧？

其实我就是为了把各种情况演示一下啦，所以做的不伦不类的，你们可不许这样，要认真哦！困，我先去睡会。

> 本章学习卡片：卡片 31
>
> 本章代码下载：本章代码

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> alay9999@163.com （刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发： http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书： http://www.jianshu.com/users/d37eaf3de0ff
- 站酷： http://www.zcool.com.cn/u/12927742

第三十二章 一些格子

然后我们来做一些不一样的东西，一些格子，当然其实他未必显示成格子的样式，但是随便他怎么掩饰，他还是改变不了在我眼中那格子的本质。

比如现在呢，我们在上面通过海报把我们这并不存在的 APP 的大特性都展示出来了，但是我们还是觉得说服力不是很够啊，所以我们要在下面横排几个小特性，一般就是有个标题，有些文字说明，再多就是加个相关的图标。

现在我们就来写这一部分。首先，依旧是他的 Html ，写出框架和内容。

```
<div id="boxes">
    <div class="box">
        <h3>我是第一个盒子</h3>
        <p>是的，没错的，十分确认的，我是第一个盒子。</p>
    </div>
    <div class="box">
        <h3>反正我就是个盒子</h3>
        <p>是的，没错的，十分确认的，我就是一个盒子。</p>
    </div>
    <div class="box">
        <h3>我是盒子我骄傲</h3>
        <p>盒子怎么了，你不能瞧不起盒子的，我们也有自尊。</p>
    </div>
</div><!-- #boxes End -->
```

然后懒得去看效果，用鼻子想也知道什么德行，一行一个呗，你可以试一下，我不拦着你，反正试试不要钱，还能加深理解。最可怕的就是你们这些人啊，跟我别的没学，就学会了我的懒，根本不想动手，那怎么学得会啊。你看我虽然懒，但是写教程还是不含糊的嘛~~（自我夸奖一下感觉真好）

现在开始猜着写 CSS 样式哦。其实吧，想想也知道 #boxes 跟 .post-content 差不多啦，都是一个 960px 宽，居中对齐的区域，用来装一些内容。

所以：

```
#boxes {
    width: 960px;
    margin: 0 auto;
}
```

为什么没有高度啊？因为我也不知道他有多高啊，其实我管他多高呢，让内部元素把他撑起来就好了。

然后是什么？横排横排，那些 .box 要横排的，然后呢？横排之后你的给个宽度啊，没有宽度他就根据元素内容自动宽度了。

```
#boxes .box {
    float: left;
    width: 300px;
    padding: 10px;
}
```

关于这个宽度很多同学不会计算，其实很简单啊，960px 的宽度，平分三份，每份就是 320px，所以每个 .box 所占的宽度就是 320px。但是我们不能让他们紧紧贴在一起啊，所以就给他们加内补（外补也行），但是这个内补的宽度也要从这 320px 里出，所以左右两边各 10px，中间的内容宽度就只剩下 300px 了。至于垂直方向上，我们对高度没什么限制，所以就不需要这么麻烦的计算了。

现在终于可以去看看效果了，然后我们再根据效果做各种修饰。虽然我做的并不好看，但是每次看效果之前还是有一种莫名的兴奋啊~



横排的效果已经如我们所愿了啊。现在就是美观度的问题，这个问题很简单，也很复杂，好辩证的讲法。所以......反正我是没有什么审美的，简单设置一下好了。

然后我就给 CSS 加上了如下内容：

```
#boxes .box h3 {
    font-size: 24px;
    font-weight: 300;
    text-align: center;
}
```

然后看一下效果：



还算看得过去，当然底下的描述文字再多几行看起来就更舒服了，我就不加了，当然你要是觉得这几个格子之间的距离太近的话，你也可以把前边我们设置的内补适当进行调整，嗯，修改数值多做实验，是有助于加深你的理解的。

然后呢，我们那个审美奇葩的老板说看着太空了，要再加点什么图，看起来更加高大上一些，所以我们来试试给每个格子都加上一个图标。

```
#boxes .box {
    float: left;
    width: 300px;
    padding: 138px 10px 10px;
    margin-top: 30px;
    background: url(../images/box-icon.png) center 10px no-repeat;
}
```

也没什么复杂的，其实前边都讲过，注意让背景居中，不重复，然后在元素顶部增加内补给图标留出位置就好。看一下效果：



当然其实同理的，你也可以把图标设置在底下。比如改做：

```
#boxes .box {
    float: left;
    width: 300px;
    padding: 10px 10px 138px 10px;
    margin-top: 30px;
    background: url(../images/box-icon.png) center bottom no-repeat;
}
```

效果如下：



当然了，你也可以给他们分别设置图标，或者用 img 标签也是完全可以的，我就不一一举例了，不过你最好都动手试一试哦。

然后我们丢掉了一个什么？清除浮动！浮动用过要清除，新手一定要记得，否则很容易造成奇葩问题让自己迷惑。去加一个 `<div class="clear"></div>` 吧，加在哪里的问题大家可以下载源代码来研究啦~

> 本章学习卡片：卡片 32
>
> 本章代码下载：本章代码

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com**（刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发：http://coffee.zji.me/

- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书： http://www.jianshu.com/users/d37eaf3de0ff
- 站酷： http://www.zcool.com.cn/u/12927742

第三十三章 一个页尾

其实写到这里，页尾大家应该都会做了的，还是那点知识而已，我都懒得写了，而且我猜性急的朋友应该也都写出来了。算了，虽然很无聊，但是还是写一遍好了。

老板说就这么多内容吧，然后你觉得如释重负啊！可是呢，就算这样也不能完啊，就算是虎头蛇尾总也是要给他一个结尾的啊。

其实结尾跟咱们的导航部分有点类似啦，都是一个通栏，然后中间一个居中的区域用来放内容啊。所以嘛，到了后面简直就是没什么新鲜的东西的啊。

来，我们先来写一个简单单的催眠一下。反正我也已经很困了啊

好啦，依旧是千篇一律的先来点 Html（糊涂妹啦）：

```
<div id="footer">
    <ul>
        <li><a href="#">公司简介</a></li>
        <li><a href="#">关于我们</a></li>
        <li><a href="#">联系我们</a></li>
        <li><a href="#">服务条款</a></li>
        <li><a href="#">帮助中心</a></li>
    </ul>
</div><!-- #footer End -->
```

内容是我瞎写的，不用在意，这个不要我去解释了吧。其实很简单的说。然后我们去定义一些 CSS （彩色式）

```
#footer {
    background:#F3F3F3;
    border-top: 3px solid #EFEFEF;
}
#footer ul {
    width:800px;
    margin: 0 auto;
    padding:15px 0;
    overflow: hidden;
}
#footer ul li {
    float: left;
    margin: 0;
    width: 160px;
    list-style: none;
    text-align: center;
}
#footer ul li a {
    text-decoration: none;
    color: #333;
}
```

到了这里我就不去过多的解释 CSS 了，因为我们用了很多遍了好吗？你应该差不多看懂了，否则大家该怀疑我故意占篇幅换稿费啦~（虽然并没有什么稿费……哭……）只是说一下：#footer ul 中的 `overflow: hidden;` 是用来清除浮动的，好啦，你又学了一个小小的新技能哦。来看看效果吧：



就是这么的简单，这根本就是前边的内容啊，分明作者……还是骗稿费的……（然后真的并没有稿费啊！）

当然这些都不要紧，关键是，要是老板看了这些教程，会觉得果然就是个骗工资的，于是责令整改啊，要做一个高大上的，跟那谁家小谁的网站一个样式的页尾……

不是说好不改了的嘛，哎，谁叫人家是老板捏！捏！，你懂的，说到捏，你有木有觉得很解气啊！但是还是要把工作做掉的，所以……前边的推翻重新来~

```
<div id="footer">
    <div class="footer-content">
        <ul>
            <li><a href="#">公司简介</a></li>
            <li><a href="#">关于我们</a></li>
            <li><a href="#">联系我们</a></li>
            <li><a href="#">服务条款</a></li>
            <li><a href="#">帮助中心</a></li>
        </ul>
        <ul>
            <li><a href="#">公司简介</a></li>
            <li><a href="#">关于我们</a></li>
            <li><a href="#">联系我们</a></li>
            <li><a href="#">服务条款</a></li>
            <li><a href="#">帮助中心</a></li>
        </ul>
        <div id="qrcode">
            <img src="images/qrcode.png" alt="">
        </div>
    </div><!-- .footer-content End -->
</div><!-- #footer End -->
```

别，别在意内容，我已经比刚才更加的随意了......嘘，老板现在不在身边。

至少，内容丰富了不是？那就好了，我就是糊弄下老板，你们......可别糊弄自己哦！

然后是 CSS ，其实，还是那点东西......

```css
#footer {
    padding: 30px 0;
    background:#F3F3F3;
    border-top: 3px solid #EFEFEF;
}
#footer .footer-content {
    width: 960px;
    margin: 0 auto;
}
#footer ul {
    float: left;
    width:300px;
    margin: 0;
    padding:0;
}
#footer ul li {
    padding:10px 0;
}
#footer ul li a {
    text-decoration: none;
    color: #333;
}
#qrcode {
    float: right;
}
#qrcode img {
    width: 250px;
    height: 250px;
}
```

然后你让我给你们解释什么？哪一句没学过？你指出来，我马上就划掉！来看看效果~



然后这章就结束了哈！但是你要自己动手试试哦~我一个页面都讲完了，你们也该写出个什么页面了吧？？记得交作业哦，虽然我不看，你们先努力把自己的页面写好写漂亮，后面我会告诉你们怎么把他们发布到网上去，跟你的女盆友们炫耀......虽然她们其实更在乎怎样把自己的照片修的美美哒~

好了，下一章我们讲点新鲜玩意儿~

> 本章学习卡片：卡片 **33**
>
> 本章代码下载：本章代码

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com**（刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发：http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书：http://www.jianshu.com/users/d37eaf3de0ff
- 站酷：http://www.zcool.com.cn/u/12927742

第三十四章 一些补遗

在开始的时候有一件事情我们一直说后面再说后面再说，但是后来我忘了......为什么酱紫......

不过现在来讲一点也不晚啊，我们来讲一讲导航里那个奇葩的链接地址的问题，回过头来改一下导航里的链接地址哦......

等等......为什么是改......因为我后面没按着那个写啊......额，我说我写错了你能原谅我吗？算了，我就这么写了，你来咬我呀！

```
<ul id="nav-items">
    <li class="nav-item"><a href="#">首页</a></li>
    <li class="nav-item"><a href="#post">下载</a></li>
    <li class="nav-item"><a href="#post-a">特点</a></li>
    <li class="nav-item"><a href="#post-b">关于</a></li>
</ul>
```

有没有觉得似乎看懂了什么？诶，链接到的好像是我们页面中的的 ID 耶。嗯哼，挺好理解的吧？这个写法叫做锚点，可以链接（用来跳转）到页面的指定元素。事实上他可以用来跳转到指定页面的指定元素。比如写作： `<a href="http://coffee.zji.me/index.html#footer"></a>`

这个效果大家可以去试试看，我就不截图了，毕竟静态图片也不是很好表示。

然后加点小料，因为老板端着他那杯 82 年的普洱过来说：要动起来，动次打次的节奏啊~~我的滑板鞋......

老板渐行渐远，我的内心却久久无法平复。还没学到 Javascript 啊！怎么做动态？好在我们还有别的办法进行变通。现在我们就来给高大上的二维码部分做一个简单的动态效果。

首先我们分析一下：我要做的是鼠标滑到二维码上二维码放大的效果。那么这个效果中有一个小要点，就是放大前后的图片哪个点的坐标是不变的？这是一个放大的基准点。假设是图片的左上角位置不变，那么放大的时候就是图片向右侧和下方进行延伸（拉伸）效果。当然这个样子的话可能显得比较突兀（毕竟我们现在还没办法做过渡效果），所以我们选择中心点不动，放大时图片向四周延伸的效果，这样视觉的焦点基没变，会觉得舒服一些。

其实 Html 部分我们完全不需要修改，只要在现在的 CSS 文件上做些小手脚就可以了的。

```
#qrcode {
    float: right;
}
#qrcode img {
    width: 250px;
    height: 250px;
}
```

这是当前相关的 CSS，因为现在的图片已经很大了，其实我们并不希望他更大，所以我们可以先把他弄的小一点，把放大的空间也预留出来。

```
#qrcode {
    float: right;
}
#qrcode img {
    width: 210px;
    height: 210px;
    padding: 20px;
}
```

现在图片内容的宽高各缩小了 40px，但是因为加了内补，所以它所占的面积并没有变化，他的中心点也依旧在原来的位置。然后我们给图片加一个 `:hover` 状态。

```
#qrcode img:hover {
    width: 250px;
    height: 250px;
    padding: 0;
}
```

其实就是鼠标滑到上面的时候恢复到以前啊，来仔细看一看，认真想一想，是不是很简单的一件事情？其实就凭一个 hover 就可以做出很多好玩的效果啦~

然后学到这里你的写的怎么样了呢？是不是比我的高大上许多倍？想不想发出来给大家看一看？下节课来教大家如何发布页面到互联网上。

> 本章代码下载：本章代码

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com** （刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发：http://coffee.zji.me/

- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书： http://www.jianshu.com/users/d37eaf3de0ff
- 站酷： http://www.zcool.com.cn/u/12927742

第三十五章 随便聊聊

现在呢，我们也做完了一个网页了，如果你恰好看懂了这些，也跟着做下来了。那么你就拥有了一个网页。挺开心的事情。但是只是自己看着这个网页你总觉得无法满足自己期望炫耀的心情。那怎么办？——发到网上去！

最简单的办法好像就是新浪的 sae（http://sae.sina.com.cn/） 了，注册开通之后，创建新应用



大概如下图填一下，二级域名和应用名称其实都是随意的。

*二级域名（App name） easycodebook .sinaapp.com 仅允许由数字，字母组成，唯一标识，也是二级域名前

注册成功后，您注册的 Appname.vipsinaapp.com 同样可以使用

建议使用独立域名

1.避免域名的搜索引擎权重受到其他用户的应用影响

2.避免了DNS劫持与污染问题

3.可以更灵活的管理应用，增强冗余性

*应用名称 我的应用 应用的中文名称，供显示用。

*验证码



应用描述

开发语言 PHP 5.3 PHP 5.6 Python 2.7 Java 1.6

全部应用和框架

最下面没显示出来的部分是选择程序开发语言，你就选择 php 空应用就行了，然后继续。之后就返回了你的应用列表，点击你那个应用的名称进入管理面板。

应用管理 我的应用（5）

创建新应用 导入应用 什么是"访问量PV"?

语言	应用信息	状态	云豆消耗（近30天）	访问量PV（近30天）
php	我的应用	正常	0	0

注意下图的打红框里是你这个应用的网址，用来访问的哦。然后点击代码管理。

easycodebook 修改应用基本信息
web应用

应用管理
应用首页
应用设置
成员管理
代码管理
管理记录
开发与调优
预定义变量
AppConfig
XHProf
应用自排查

近30天访问量PV 0
近30天云豆消耗 0颗
购买云豆 设置预算

二级域名 easycodebook.sinaapp.com
easycodebook.vipsinaapp.com
独立域名 尚未绑定 绑定

绑定独立域名可以避免域名的权重受到其他用户影响，避免DNS劫持

然后建立一个版本，按钮在右上方，版本号是 1 就行，创建后界面如下

代码管理

创建一个版本

已用:0.0M/100M

代码空间扩容

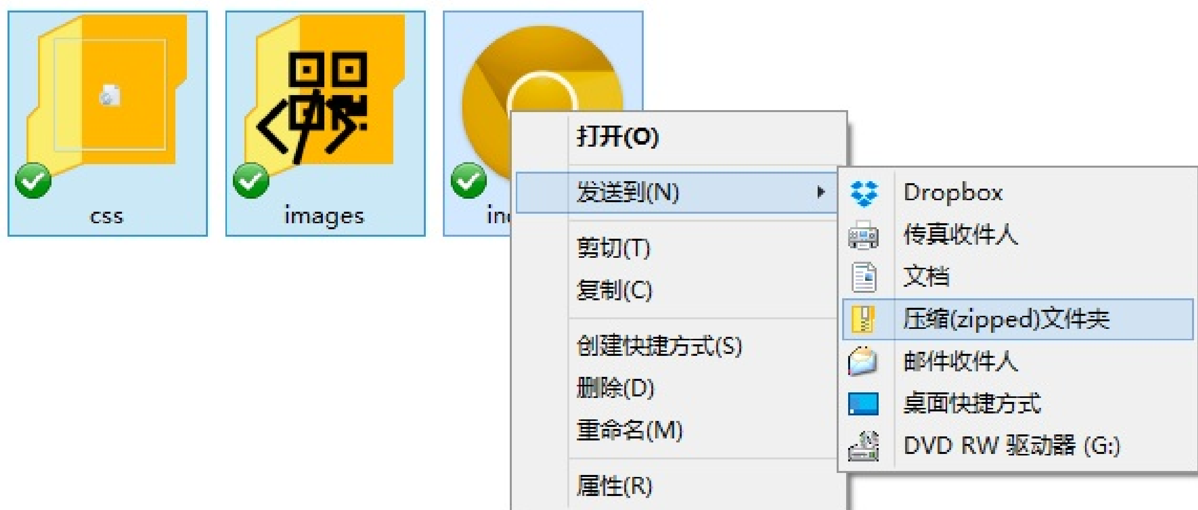
默认版本	部署时间	链接	xhprof调试	错误显示	代码编辑	操作
1	2015-03-20 20:03:52	http://1.easycodebook.sinaapp...	开启1小时	关闭显示	编辑代码	操作

上传代码包

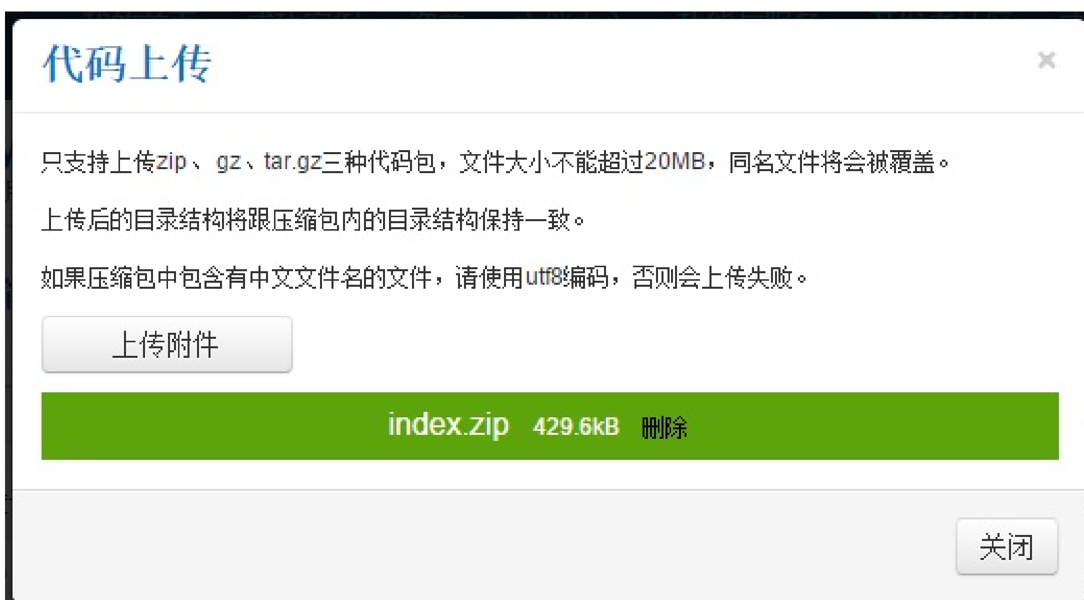
删除

然后点击右侧操做按钮，选择上传代码包（如上图）。这个代码包就是把我们的代码打包为 zip 格式，对，用你的压缩软件就行。记得把你的网页命名为 index.html

打包的时候要从 index.html 文件这一层开始打包，外面别再套着文件夹了。就像下面这样打包。



然后这个压缩包名字用个简单的英文或者数字，免得出意外。上传上去，等到上传结束，如下图



点击关闭，然后用前面记录下来的网址就可以访问了。

然后由此开始讲一些网站的基础知识，做网站我们需要有三样东西：域名、空间、程序。

域名：上面我让你记下来的那个网址就是域名，当然也可以自己购买个新的域名，这就相当于你的家庭地址

空间：用来放网页的地方，空间会有一个 IP ，就相当于你家所在的经纬度，我们需要通过一个操作（绑定域名）来把你的域名指向到这个 IP，才能在访问域名的时候访问到这个空间的内容。

程序：比如你写的这个网页。

也就是通过域名这个住址访问到你家里的你。然后我们一般通过一些 FTP 软件去管理空间上的程序，或者说是管理那些既存在服务器上的文件。域名和空间都可以在网上购买，当然空间也被称作虚拟主机。

网站空间的的管理面板，可以用来管理 FTP 账户，绑定域名，数据库，甚至空间中的文件。比较常用的有两种：DA（DirectAdmin），CP（cPanel），当然在国内买应该都是中文面板，其实你只要在其中找自己需要的项目就行了，不懂得不管他。

这样我就给了你很多的关键词，如果你有这方面的需求可以去搜索相关的教程，在这里我就不详细地去讲了。

说说版本管理，常见的有两种 SVN 和 Git 。前面说的 SAE 用的是 SVN 来管理代码。Git 则有著名的 Github.com （一个著名的开源代码托管网站）。

版本管理的概念是这样的（通俗版），你写了一个文件，然后保存为版本 1，然后你修改一下这个文件，保存为版本 2，然后依此类推，到了版本 200。这时候我们需要找到版本 23 的内容......这个案例作为设计师一定觉得心头在滴血。版本管理在这时候就可以非常方便的找出来。

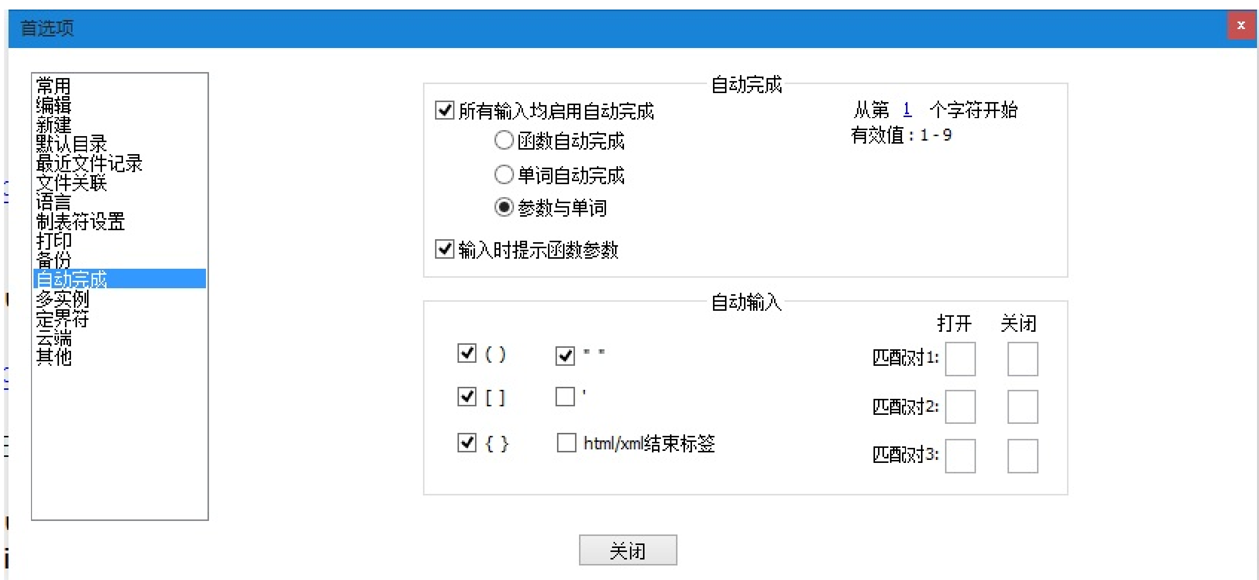
然后还可以分支：就好像我要对这个图片做一些不一样的尝试，所以我先把她复制一份去实验，嗯，这就是分支但是当实验成功了你还可以合并分支......

其实版本管理对于设计也是很有用的东西，只是很少有设计师愿意去了解一下这方面。

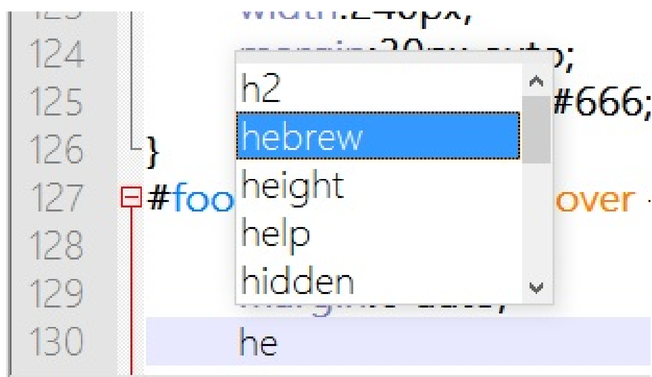
本书就是用 git 进行版本管理的。

关于代码自动完成的一点说明：为啥不用记代码？有自动完成谁还记他，见到认识就很好了。

Notepad++ 默认是开启的，如果没有可以到 设置——首选项——自动完成 中开启一下



开启之后，只要已经选定代码的语言（菜单栏，语言），当然保存为对应的后缀也算选定了语言，那么写代码的时候就会有提示



就像上图，我写 he 的时候就已经有 height 的提示了。然后上下键选择到你需要的项目，Tab 或者回车键都可以。这样写代码的效率是很高的。

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com** （刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发： http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书： http://www.jianshu.com/users/d37eaf3de0ff
- 站酷： http://www.zcool.com.cn/u/12927742

第三十六章 定位（一）

前面我们写了一个网页，当然，我们都知道这是一个很简单很基础的网页。只是我们心照不宣的照样拿他出去炫耀罢了，咳咳，这个秘密谁也不许往外说，否则莫怪为师......

串词了，没事，你就当没看见。那么大家看前边的布局的时候有没有觉得很单调乏味呢？想横排，用浮动，浮动之后要清除；想居中，两边补，外补自动有宽度。就这点东西吧，翻来覆去还是这点东西。很无聊的，也很死板的，仅凭这样的三脚猫功夫肯定是搞不出来灵动洒脱的网页来的。

当然了，要是就是某些网游的官网，坚持十年前的套路，十年了，那些结构都没变过，换换背景就是另一家游戏......那你现在的水平应该可以应付了的说。我在说什么你懂的，反正我是不懂的。

得，咱们闲言少叙，书归正文，啪，响木这么一拍呀，别的咱不夸......

那个这一节开始我们讲一些元素的定位方式，其实吧，他们说什么绝对定位相对定位的我也分不清，总觉得有点故弄玄虚。但是我知道每一种定位方式是怎么回事，什么时候该怎么用。反正到目前为止~~一只狗用~~一直够用，除了没法装[哔~]。

元素定位方式的属性是：`position`，他的默认值是`static`，这个了解下就好，我又是现查的，因为并不常用啊，默认值一般都省略不写了。

现在我们来认识几个特殊的定位方式：

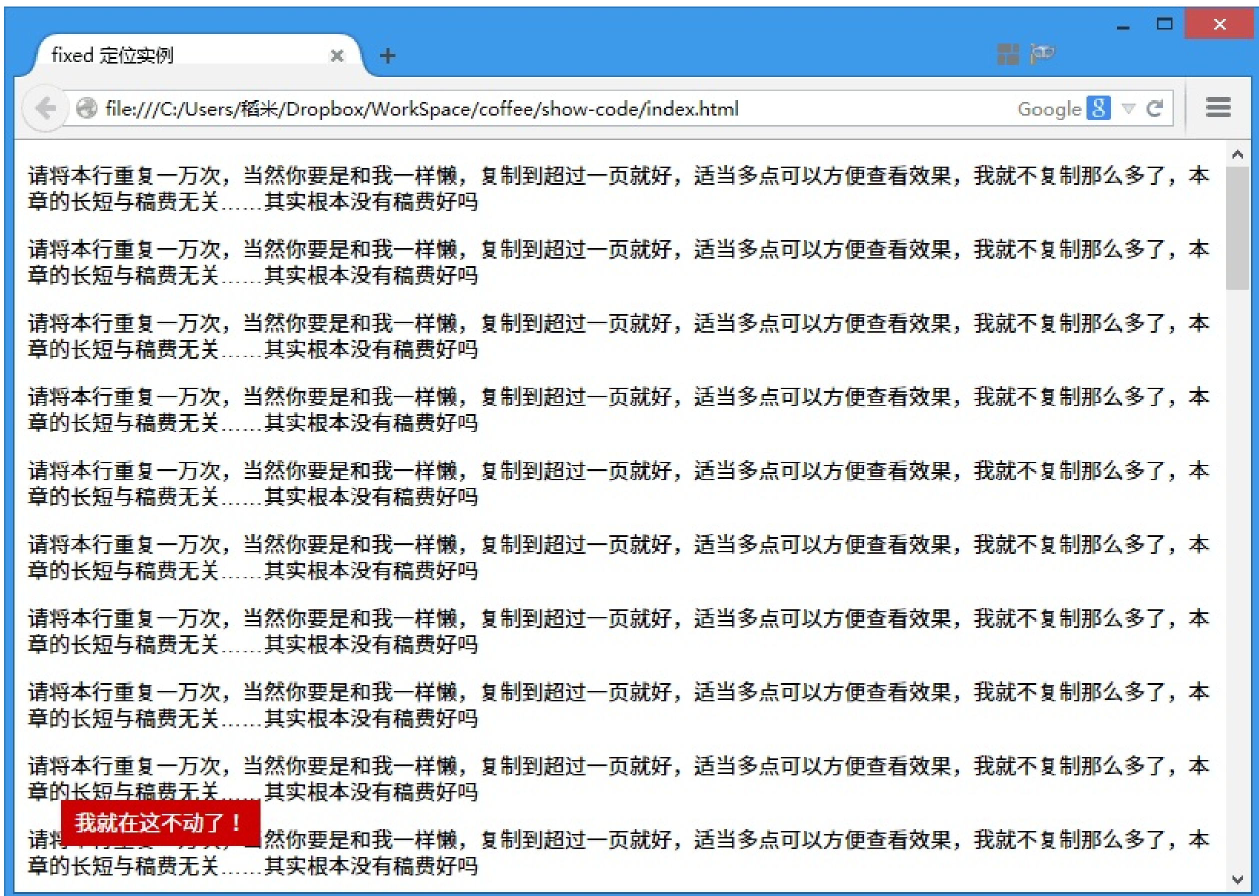
fixed，这个应该是最受新人欢迎的定位方式，好理解，效果明显，也好玩。他是相对于浏览器的定位。

就是你随便怎么上下滚动页面，这家伙就铁了心的不动弹。因为他是相对于浏览器的窗口的，浏览器窗口没移动和变化，那其他的事就跟他没啥关系，这就好像十字路口的交警，任你车来车往，他就站在那里坚守岗位。

然后我们举个简单的例子说明一下：

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>fixed 定位实例</title>
    <style>
        #miao {
            width: 130px;
            height: 30px;
            position: fixed;;
            left: 30px;
            bottom: 30px;
            background: #C00;
            line-height: 30px;
            color: #FFF;
            text-align: center;
        }
    </style>
</head>
<body>
    <p>请将本行重复一万次，当然你要是和我一样懒，复制到超过一页就好，适当多点可以方便查看效果，我就不复制那么多了，本章的长短与稿费无关……其实根本没有稿费好吗</p>
    <div id="miao">我就在这不动了！</div>
</body>
</html>
```

试试效果吧，你怎么滚动那个红色的元素总在左下角，不会发生移动，如下图：



那么上面的代码大部分都很容易理解了，我来解释一下其中的 `left` 和 `bottom` 。我们说： `position: fixed;;` 是说定位方式，相对于浏览器窗口定位。但是我们说他的位置了没有？没有啊！

这 `left` 和 `bottom` 就是说明他的位置的，我解释给你听啊： `left` 就是说元素的左边缘和浏览器的左边缘之间的距离；同理 `bottom` 就是说元素的下边缘和浏览器的下边缘之间的距离。

然后聪明的你就可以举一反三地说出 `right` 和 `top` 的意思了。嗯，这四个东西下节课还要用哦~

留个作业，除了那种悬浮的“在线客服”或者广告之类的东西以外，大家找找 fixed 定位还在哪里被用出了高大上的效果？

> 本章代码下载：本章代码

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com** （刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发： http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书： http://www.jianshu.com/users/d37eaf3de0ff
- 站酷： http://www.zcool.com.cn/u/12927742

第三十七章 定位（二）

上节课我们讲了 fixed 定位，那么现在我们来简单回顾一下他。fixed 定位就是锁定元素和浏览器的相对位置，无论怎么滚动页面，这两者之间是相对静止的。然后还有一个细节上一节我们没有注意哦！

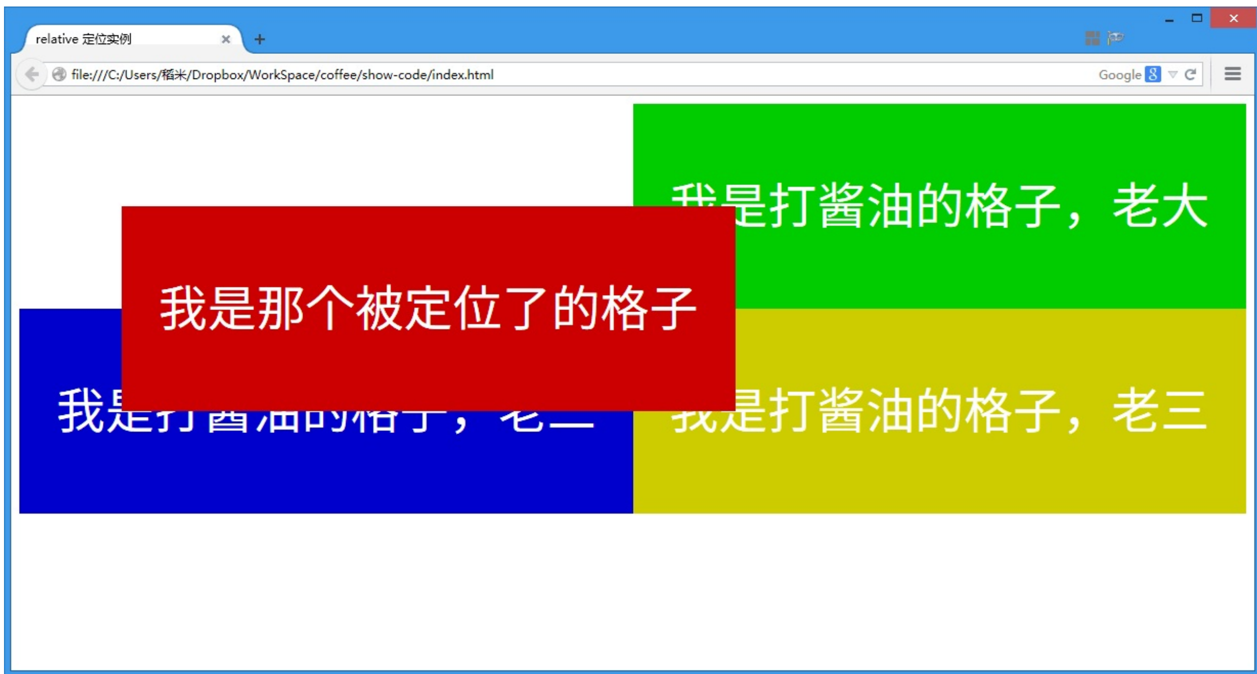
fixed 定位的元素不在文档流内！这只箱子从传送带上拿出来了，挂在浏览器上一动不动的。所以他和传送带上的箱子互不影响，传送带上也没有他的位置。

然后再来认识一个新的定位方式：`relative`，不解释，先看例子再说话。

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>relative 定位实例</title>
    <style>
        div {
            width: 50%;
            height: 200px;
            float: left;
            color: #FFF;
            font-size: 48px;
            text-align: center;
            line-height: 200px;

        }
        #a {
            background: #C00;
            position: relative;
            left: 100px;
            top: 100px;
        }
        #b {
            background: #0C0;
        }
        #c {
            background: #00C;
        }
        #d {
            background: #CC0;
        }
    </style>
</head>
<body>
    <div id="a">我是那个被定位了的格子</div>
    <div id="b">我是打酱油的格子，老大</div>
    <div id="c">我是打酱油的格子，老二</div>
    <div id="d">我是打酱油的格子，老三</div>
</body>
</html>
```

这些代码不多说，都知道怎么回事，除了这个定位方式，看了效果我们来说。



如果没有定位，就是这四个格子排排坐，但是不给果果吃，所以无聊的很。但是我给第一个元素做了 `relative` 定位，这是啥意思呢？就是叫这排排坐的某一位小朋友出来唱个歌。唱完他还回去坐，对吧，所以他的位置要留着，不能他出来唱歌别的小朋友就坐过去，要不然唱完歌回来发现没位子了要哭的。所以你看后面的小朋友还都坐在那里，虽然前面有了一个空着的位置。

然后呢，去唱歌的小朋友站在舞台上，他不会影响到坐着的那些小朋友，虽然我们看到他挡住了后面坐的的小朋友，但是只是视线上的遮挡，并不会发生什么肢体上的冲突，否则……去看演唱会什么的也太爽了点。

然后就是位置问题，其实也挺简单的，`left` 就是现在位置的左边缘距离原来位置的左边缘的距离。就是老师说，小红，来，起来给大家唱个歌，从你座位开始，向前走两步，再向右走三步，站在那里唱（这个老师是个逗[哔~]）。那么小红唱歌的位置跟她坐着的时候左手移动的距离是三步，而后背移动的相对距离是两步，这都很好理解。

那么，现在我们知道了 relative 的定位方式，在文档流中保留原有位置，然后基于原来的位置进行偏移。现在仔细想想，跟 fixed 的区别还是蛮大，很好记。但是跟下一节要学的就容易混淆了，所以先把今天的记牢再看下一节。

> 本章代码下载：本章代码

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com**（刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

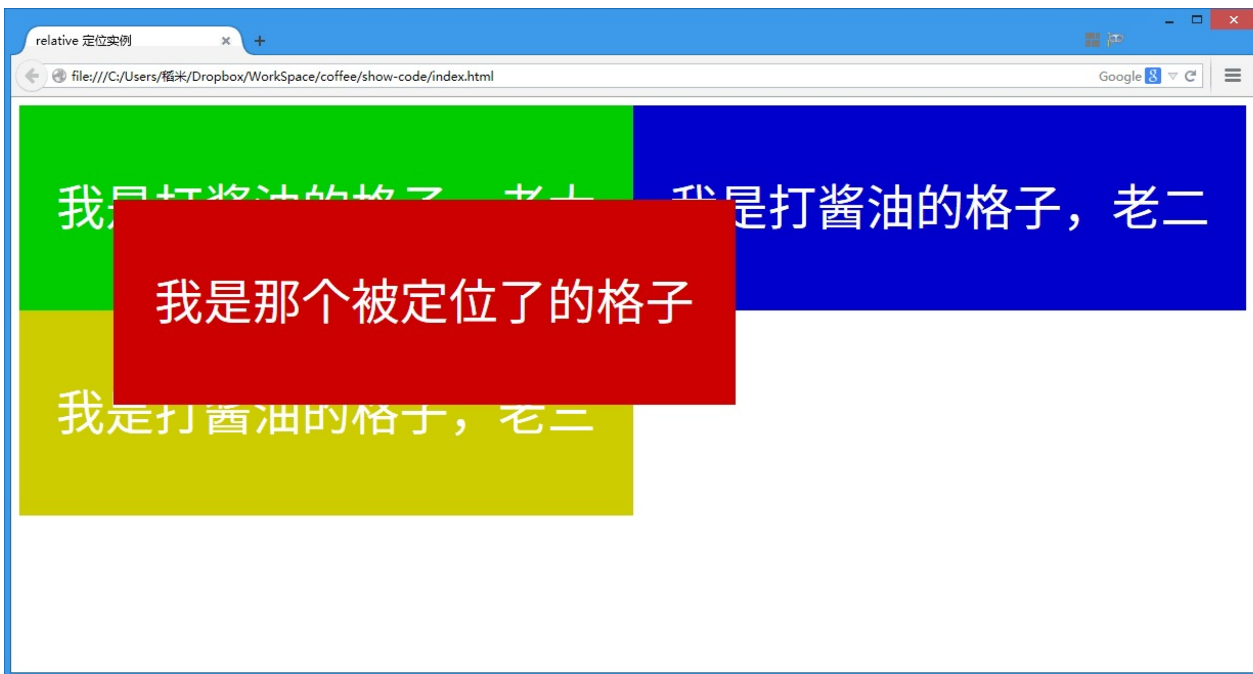
- 主站首发： http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书： http://www.jianshu.com/users/d37eaf3de0ff
- 站酷： http://www.zcool.com.cn/u/12927742

第三十八章 定位（三）

前边的都记住啦？那就来看今天的。

其实我觉得，日常用 `relative` 定位的地方并不多，但是又绝对不少。这话怎么理解呢？就要先看今天的定位方式：`absolute` 了。

这个简单，就把上节课的例子的定位改成 `absolute` 就好了，我就不再复制一遍了，来看看效果。



现在看到，红色的格子没了他原来的位置。这就是老师说："小明，你给我搬着桌子椅子滚出去！"然后就连他的位置都不给留了，就当没他这回事。

然后再说定位，老师都不要小明了，所以他也不可能相对于他原来的位置进行定位了，班里都不承认他了，哪里还有位置啊。

那么他的 `left` 之类的属性相对于谁呢？相对于套在他外面的，从他开始往外数，第一个定位方式不是 `static` 的元素。这个听起来乱，其实也很简单，就是不是默认定位方式的元素。

就是小明这孩子虽然淘气，但是有良心啊。默认的不敢跟别人说自己是被某某学校或者说自己被某某班开除的，他怕给学校啊、班级啊......丢脸。只有当这些集体声明过不在乎（声明其他的定位方式），他才敢说自己是从那里出来的。而且，为了牵连更少的人跟自己丢脸，如果班级和学校都做了声明，他只说班级，不说学校。

如果他外面的元素都没有声明其他的定位方式怎么办？那他相对于页面的最边缘进行定位，注意是页面，不是 html 或者 body 元素，是更外面的页面边缘。

但是这样想找小明在哪里真的好难！所以一般的我们会给父级元素进行 `relative` 定位，但是不进行偏移，就是不设置 left 什么的值，这样父级元素还在他原来的地方，还站着原来的位置，仿佛什么都没变化过。这就明白了为什么前边我说 `relative` 用的不多却也不少，因为单独用它本身特性的地方可能比较少，但是出现 `absolute` 定位的时候，他一般都会在外层辅助性的出现。

就是小明被轰出去了，老师很开心，说我们全班大合唱庆祝一下吧，但是不用动地方了，都在座位上唱就好了。反正我说的挺玩笑的，你理解就好，后面我们会有案例来专门说明这些问题的。现在就是把故事听明白，知道我比喻的是些什么。

> 本章代码下载：本章代码

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com**（刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发：http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书：http://www.jianshu.com/users/d37eaf3de0ff
- 站酷：http://www.zcool.com.cn/u/12927742

第三十九章 坐标系

现在有些同学可能已经有疑问了，你看这几种定位啊，都是让元素到了别的元素的前面。但是要是我们用这种方式把多个元素定义到了相同的位置，究竟谁在前面谁在后面？

这时候就涉及到了网页的坐标系，这是一个三维的坐标系，然而并不复杂在三维层面。真正让人头疼的是二维上的各种变化。让我们从最简单地聊起。

最简单的，网页的的左上角为坐标系的原点，向右为 X 轴的正方向，向下为 Y 轴的正的方向。那么向着屏幕外则是 Z 轴的正方向。

X、Y 轴我们用诸如像素之类的单位进行测量，而 Z 轴则只有顺序值，没有单位，它对应一个单独的属性值 `z-index` 来表示元素在这个方向上的顺序，这有就是我们前边问题的答案。（这个下一章进行演示）

如果只是这样我们当然很容易理解和掌握，但是很多时候事情没这么简单。元素的位置并不是固定的，所以有些东西就很难用默认的原点进行计算了。比如前边学习的背景位置（ `background-position` ），我们不能说元素背景在上边这个默认坐标系中位置，那样计算很麻烦，而且我们以后如果需要给元素移动位置，还要再次计算这个坐标值，显然很烦恼的。所以我们就用了相对坐标系。以元素的左上角为原点，去计算背景图片的偏移量（坐标）。那么其实这个背景定位的值也就是背景图的左上角相对于原点（元素的左上角）的坐标值。

你看我说了这么多左上角，因为在网页中左上角是一个很常用的定位基准，为什么选他？因为你的元素宽高的变化并不会影响左上角的位置（大部分情况下）。这样比较便于计算，这也是希望让事情变得简单。其实元素的宽高也可以看作是元素右下角点相对于左上角点（原点）的坐标。

然后再说前边用到的 `left` 、 `right` 、 `top` 、 `bottom` 这几个属性。他们涉及到的坐标系其实只有一个轴，比如 `left` 它涉及到的方向就是左右而已，或者说只是 X 轴的正负方向而已，这时候我们再去强调是哪个轴就已经是画蛇添足了。然后如果我们还按照基本的坐标系去判断正负方向依旧是徒增烦恼的做法，所以干脆根据我们的第一直觉，向着元素内的方向即为正方向，这样好记好判断。

同理的，padding 的四个方向也因为相同的原因而视为向内的是正方向。

margin 是和 padding 相对的，他们补的方向相反，所以 margin 以向元素外的方向为正方向。

这一章文字不多，但是很重要，因为这些问题几乎遍布了元素定位的每一个方面，希望仔细想想，努力去理解他们。

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com**（刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发：http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书：http://www.jianshu.com/users/d37eaf3de0ff
- 站酷：http://www.zcool.com.cn/u/12927742

第四十章 定位实例（一）

讲了这么多，有点晕了？嗯，很好，很好！来我们做个练习来更加的迷糊一下，咳咳，是深入理解一下。

其实这个案例也十分的简单，本来是可以在前面所写的页面中直接添加的，不过怕大家搞乱了，所以索性新写一个好了。当然我们只写主要部分，不会再写一个完整页面的。

我们要写的是一个二级导航，和一个返回顶部的按钮。其他部分咱们就简单呵呵一下，你们要理解我在简言之。先来写个基础的结构哦~

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>定位练习</title>
    <style>
        html, body {
            margin: 0;
            padding: 0;
        }
        #main {
            width: 960px;
            margin: 0 auto;
        }
        #logo {
            height: 120px;
            margin: 10px 0;
            background: #F3F3F3;
        }
        #post {
            height: 300px;
            margin: 10px 0;
            background: #F3F3F3;
        }
        #footer {
            height: 160px;
            margin: 10px 0;
            background: #F3F3F3;
        }
        #nav {
            height: 36px;
            margin: 10px 0;
            background: #333;
        }
    </style>
</head>
<body>
    <div id="main"><!-- 页面内容，宽 960 Start -->
        <div id="logo"></div>
        <div id="nav">
            <!-- 后面的 Html 结构写在这里，因为我们要写一个导航 -->
        </div>
        <div id="post"></div><!-- 这是海报 -->
        <div id="post"></div>
        <div id="post"></div>
    </div><!-- 页面内容，宽 960 End -->
    <div id="footer"></div><!-- 这是一个页尾 -->
</body>
</html>
```

这些都看得懂了吧，你看，简单的几行一个基本的页面结构就出来了，难吗？理解了就真的没什么了。自己去看下效果，我不截图了，这些只是给我们今天的主角一个舞台而已。

然后来写个导航，也没啥大不了的，轻车熟路啊~

```
<ul>
    <li>首页</li>
    <li>分类 &#9660;</li>
    <li>关于</li>
    <li>联系</li>
    <li>福利</li>
</ul>
<div class="clear"></div>
```

他的 CSS 如下：

```
#nav ul {
    margin: 0;
    padding: 0;
    float: left;
}
#nav ul>li {
    margin: 0;
    padding: 0 35px;
    list-style: none;
    float: left;
    color: #FCFCFC;
    line-height: 36px;
}
.clear {
    clear: both;
}
```

这也全是前面学过的内容，我们不多说，链接我不写了，反正这也不是我们这次的重点，我们尽量写的清爽一些。

然后接是一个新的选择器 `>` 表示某元素的子元素里的什么。`#nav ul>li` 就是说 `#nav ul` 的所有是 li 子元素。就这么说吧，以前我们说 `#nav ul li` ，是说这个 ul 的所有后代，哪怕是七十八代玄孙，只要叫 li ，那就有一个算一个。现在 `#nav ul>li` 就只说他儿子辈里面叫 li 的，跟其他辈分没有毛关系了。

你说我们写了这么半天学过的东西，没见到啥区别啊。嗯……你看到分类那一项我写了个向下的三角箭头么？我们要给他写一个下拉菜单，不过……打字打得手疼了，下节课再继续……

第四十一章 定位实例（二）

那么我们说了，这个要做一个二级菜单的，现在刚有了第一级的菜单，现在来写第二级。

二级菜单是什么呢？其实还是一个列表，有那么几个列表项。所以还是 ul 和 li 标签而已。那么放在哪里呢？这个问题很有意思，真正的解答是：放在哪里都可以。我来解说几种情况：

第一种：像我们这个实例这么简单的情况。一般的我们为了说明他的层级关系，都会把它放在父级导航项内；

第二种：二级甚至三级或者更多级导航很复杂，代码量很大。这时候我们一般把它放在页尾。因为，一般这些导航默认是不显示的，那么让不显示的代码在前面，打开页面的时候就会先载入这部分代码，然后再载入下面那些需要被显示的代码，就显得页面加载速度比较慢了。所以我们把这些不需要立刻显示出来的代码放在最后去加载。至于位置，当然用各种方式定位过去就是了，有的时候也可能用 JS 动态的把这部分代码再插到前面。

好啦，开始我们本次的简单任务，那么加上二级导航之后就变成了这个样子。

```
<ul>
    <li>首页</li>
    <li>分类 &#9660;
        <ul>
            <li>小分类一</li>
            <li>小分类二</li>
            <li>小分类三</li>
            <li>小分类四</li>
        </ul>
    </li>
    <li>关于</li>
    <li>联系</li>
    <li>福利</li>
</ul>
<div class="clear"></div>
```

然后我们来给二级导航写 CSS ：

```
#nav>ul>li>ul {
    margin: 0;
    padding: 0;
    background:#363636;
    position: absolute;
    top: 40px;
}
```

我们一段段的说，这里定义的是 #nav （导航）里的 ul （一级菜单）里的 li （一级菜单项）里面的 ul （二级菜单）。好了，选择器就说到这里，下次不说了，说了两遍，差不多该看懂了。

margin 和 padding 都是清除元素的原有样式，这事情我们也做过很多次了，background 是设置背景色。

position 是定位，absolute 定位什么特点？不给自己保留原来的位置，而且相对于他外面第一个不是默认定位方式的元素定位。然而他外面并没有……这时候定位变得复杂。所以我们要让它相对于它外面的那个 li 标签进行定位。那么我们要给前面的 li 标签设置一个定位 `position: relative;` ，但是并不设置定位的位置数值。这样 li 位置没有发生变化，而且也占据着原来的位置。但是它内部的 ul 就可以以它为基准进行定位了。

top 是说二级菜单相对于他外面的 li 标签，顶部的距离是 40px ，正好给一级菜单留出了位置。

然后我们定义一下二级菜单的菜单项就可以去看效果了：

```
#nav ul>li>ul>li {
    width: 80px;
    margin: 0;
    padding: 0 35px;
    list-style: none;
    float: left;
    color: #FCFCFC;
    line-height: 36px;
    border-bottom: 1px solid #666;
}
```

这部分不解释了，因为并不复杂。

然后我们就获得了如下效果的导航菜单：



那么趁着还不复杂，赶紧的思考一下这个定位是怎么实现的，把你不懂得地方都去试着修改或者删除一下，看看会发生什么样的变化。

> 本章代码下载：本章代码

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com** （刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发： http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书： http://www.jianshu.com/users/d37eaf3de0ff
- 站酷： http://www.zcool.com.cn/u/12927742

第四十二章 定位实例（三）

然后我们的问题是：这货怎么一直这么显示着，这不是我想要的结果啊，要是可以鼠标划到上面才显示，鼠标离开就隐藏有多好？当然了，这种效果几乎是各个网站所必备的。

第一步：我先把它隐藏掉，省得碍眼。

这个最简单了，给 `#nav>ul>li>ul` 选择器里面加一个属性就好，`display:none;` ，意思就是显示方式是不显示。好了，不碍眼了，但是，他会一直隐藏，那就没什么作用了，所以：

第二步：鼠标滑过显示。我们说过鼠标滑过这个状态使用伪类 `:hover` 来表示。所以其实我们如下设置即可：

```
#nav>ul>li:hover>ul {
    display: block;
}
```

这里要理解的一个问题是，我们把鼠标划到哪个元素上，而显示出来的又是哪个元素？那么这里是当鼠标放在 #nav>ul>li 上面的时候，它里面的 ul 被显示出来。

那么我们预期的效果就达到了，大家要自己研究一下源码，定位这个问题在 CSS 里面是个很重要的问题。

然后我们开始下一个练习，给这个页面右下添加一个返回首页的按钮，就像很多网站都有的那样。大家应该都知道什么样子，他不随页面滚动，一直处于页面的右下角。

像这种浮于页面之上的东西，他并不占据什么位置。也很难说他在逻辑上属于什么元素。而且它的重要性也不是很高，只是增加用户体验，而不是用户打开页面所第一预期的内容。所以这种代码一般放在页尾，就是 `</body>` 标签的前面。

那么这个按钮写起来很简单的：

```
<div id="go-top"><a href="#">返回<br>顶部</a></div>
```

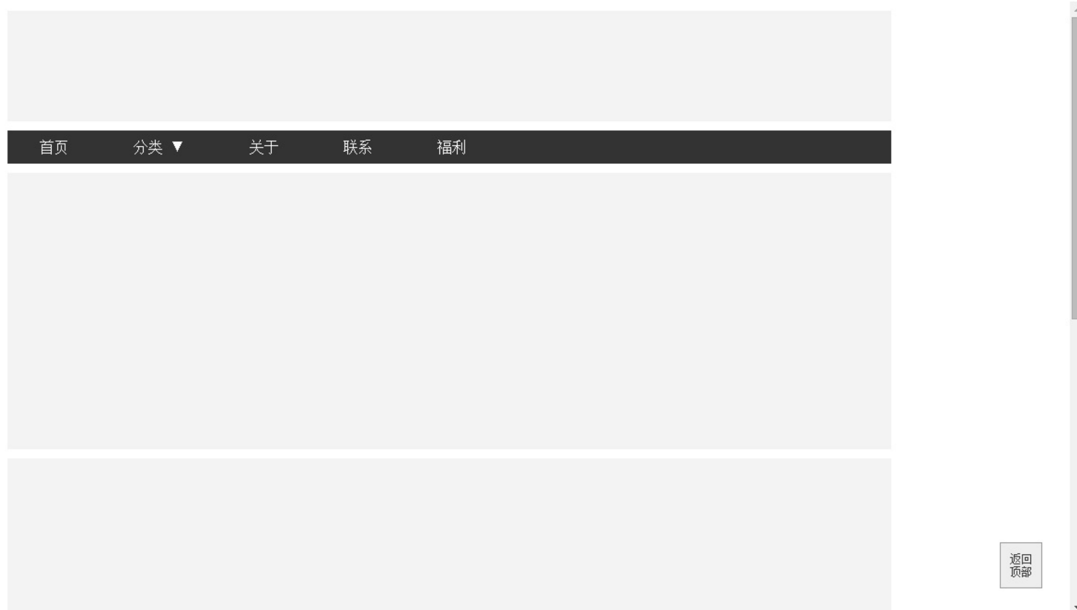
简单吧，一个链接而已，我们知道 # 可以链接到锚点，但是后面如果没有锚点名称，只是一个简单的 #，那就是链接到当前页面的顶部。其实说是滚动上去更贴切一些。里面文字什么的我就不用解释了吧。

然后给他设置样式：

```
#go-top {
    position: fixed;
    right: 30px;
    bottom: 30px;
    border: 1px solid #999;
    background: #EEE;
    font-size: 12px;
    text-align: center;
}
#go-top a {
    display: block;
    padding: 10px;
    text-decoration: none;
    color: #333;
}
```

这个太简单，不讲了。自己动手改改参数，或者把文字换成图片玩吧。下节课开始我们要给这个页面再加一些高大上的内容，哼唧~

效果见下图的右下角。



> 本章代码下载：本章代码

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com**（刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发：http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书：http://www.jianshu.com/users/d37eaf3de0ff

- 站酷： http://www.zcool.com.cn/u/12927742

第四十三章 定位实例（四）

导航我还没玩够，哼哼，我要写一个帅点的，这样我们就可以顺便学一些新鲜的玩意了。

首先左下角浮动一个按钮，哎呀，这个写起来跟右下角的一样呢~我就不多废话了，对了，区别有点，就是这次我们不用链接了。

```
<div id="show-nav"><div class="button">我<br>是<br>菜<br>单</div></div>
```

然后我们开始对它进行美化，要是还跟右侧的按钮一样，这页面岂不丑死！来来，我们来写一个长长的 CSS 样式。

```
#show-nav {
    position: fixed;
    left: 0;
    bottom: 120px;
}
#show-nav .button {
    width: 16px;
    height: 64px;
    padding: 4px;
    position: relative;
    background: #AA0000;
    border: 5px;
    border-style: solid;
    border-color: #FFF;
    border-radius: 0 16px 16px 0;
    border-left: none;
    box-shadow: 2px 2px 18px #333;
    font-size: 12px;
    color: #FCFCFC;
    line-height: 16px;
    text-align: center;
    cursor: pointer;
    z-index: 99;
}
```

呵，有点长，我们慢慢讲。#show-nav 现在只负责定位，那么他的 CSS 很容易解读了，只是简单的设置了定位方式和位置而已。

`#show-nav .button` 主要负责样式，其实大部分属性我们都是可以自己看明白的，我就不反反复复地去讲了，否则……估计有人要跟我叫“婆婆”了。

`position: relative;` 这个属性又是一个助攻，借助它使得元素可以设置 z-index 属性。而并不是用它去影响元素位置。当然也可以说他影响了元素在层级上的定位。

边框 border 这个复合属性被我拆开来写了，先完成各种边框的定义之后，又用 `border-left: none;` 把左侧的边框干掉了。其他都好理解了，然后有一个新的属性是 `border-radius` ，这个设置的是边框的圆角，可以设置一个值，代表圆角的半径，四个角的圆角半径相同。现在我设置了四个值，显然是把四个角分别设置了。四个值的顺序是：左上、右上、右下、左下，依旧是顺时针旋转，所以记住从左上角开始就够了。

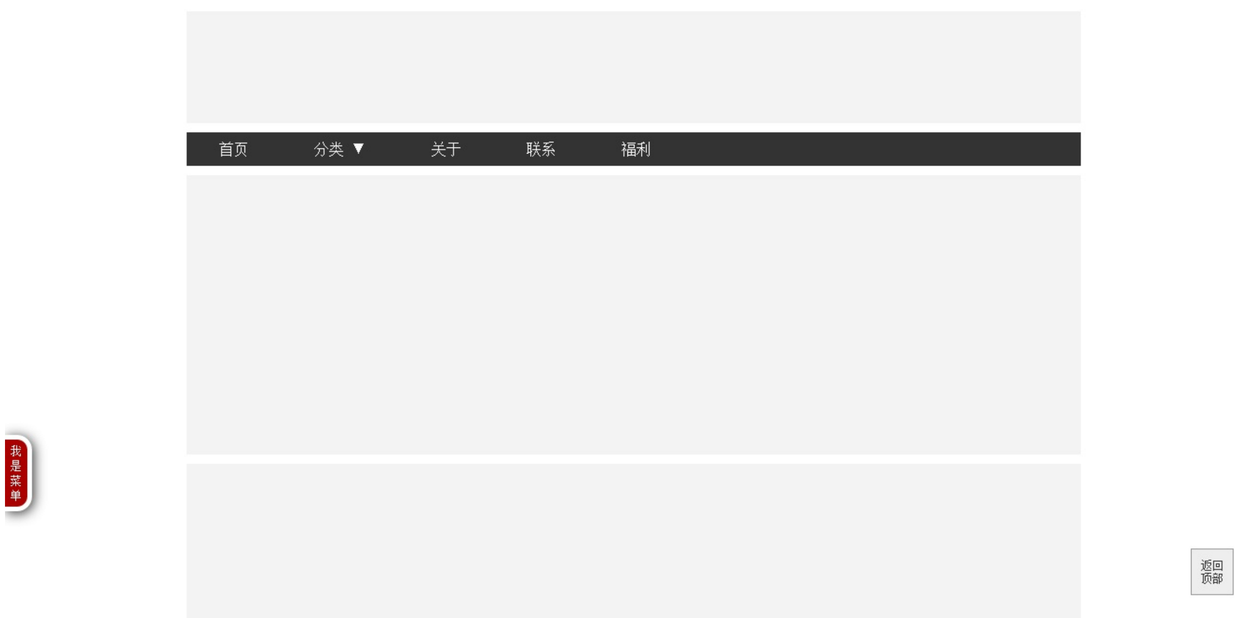
`box-shadow` 是指元素的阴影，属性是三个值和一个颜色，前三个值分别是 X轴偏移量，Y轴偏移量，阴影扩散大小，最后的颜色当然就是阴影的颜色值了。这个不懂就试一试，实践是最好的学习方法，改改数值，看看效果立刻就明白了。

`cursor` 是说鼠标指针样式，当然是限定鼠标在这个元素之上的时候，pointer 标识的就是那个小手的样子，不知道？拿鼠标指向一个链接，就是这个样子的。

剩下宽高，行高，文字居中这些确定了文字的位置，你可以想一想我是怎么计算的，如果想明白了就修改数值看看，看你是否可以把他搞到另外的尺寸而保证文字居中。

那么现在我们把一个简单的元素搞得复杂了，然而仔细想想，也并没有做什么很奇特的事情。CSS 也不过就是为每个元素统筹安排这点有限的属性罢了。

那么来看看我们左侧 Sao 红的按钮吧~



> 本章代码下载：本章代码

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com** （刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发： http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/

- 简书：http://www.jianshu.com/users/d37eaf3de0ff
- 站酷：http://www.zcool.com.cn/u/12927742

第四十四章 定位实例（五）

然后就是给这个按钮加装一个菜单用来弹出。这很像我们前边写二级菜单的操作，其实也没什么本质上的区别，管他几级呢，反正都是弹出菜单罢了。

所以也不过是加一个列表而已，懒惰如我，直接把上边的二级菜单复制过来了。

```
<div id="show-nav">
    <div class="button">我<br>是<br>菜<br>单</div>
    <ul>
        <li>小分类一</li>
        <li>小分类二</li>
        <li>小分类三</li>
        <li>小分类四</li>
    </ul>
</div>
```

这并不复杂，那么他的 CSS 跟上边二级导航的也差不多：

```
#show-nav ul {
    display: none;
    margin: 0;
    padding: 0;
    background:rgba(54,54,54,0.6);
    position: absolute;
    left: 20px;
    bottom: 20px;
    z-index: 50;
}
#show-nav:hover ul {
    display: block;
}
#show-nav ul>li {
    width: 80px;
    margin: 0;
    padding: 0 35px;
    list-style: none;
    float: left;
    color: #FCFCFC;
    line-height: 36px;
    border-bottom: 1px solid #666;
}
```

除了定位值发生了一些变化，其他几乎没有改变，所以就不多说了。只说一个小区别：

`background:rgba(54,54,54,0.6);` ，一般我们用的如 #FFFFFF 形式的颜色值，是 RGB 颜色值，第一二位表示红色的浓度，三四位是绿色，五六位是蓝色。现在我们换了一个 rgba 格式，其实就是多了一个透明度的数值，当然其他区别也是有的，#FFFFFF 用的是十六进制，而我们现在写的 rgba 后面用的是十进制，前三个数值分别表示红绿蓝三种颜色的浓度，数值为 0——255 之间的整数。而最后一位是透明度，可以是 0 到 1 之间的小数。这个办法就可以简单的设置一个透明背景。

然后注意这次设置的 `z-index:50;` 而上节课我们给 `#show-nav .button` 设置的是 99。他们数字的值并不重要，重要的是他们之间的大小关系。这样是为了让弹出的菜单显示在按钮后面，也就是按钮遮住弹出菜单的左下角，显得更美观一些。

然后来看看效果：



这几节课的案例都是很常见的一些情境，定位的用法也并不复杂，稍微认真思考一下都可以理解，然而重要的不是技术本身，而是利用技术的思路。嗯哼，多思考哦~

> 本章代码下载：本章代码

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com** （刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发： http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书： http://www.jianshu.com/users/d37eaf3de0ff
- 站酷： http://www.zcool.com.cn/u/12927742

第四十五章 无聊的写个小游戏吧

当然了，没有 JS 这种灵活的东东，我们不可能做很复杂的动态游戏。但是如果你肯开发思路的话，确实可以实现一些完整的游戏过程，虽然确实很简单，但是可玩性，以及游戏体验却也未必很差。

那么今天我们来做一个猫咪抓老鼠的小游戏。你看说的好听，其实就是不断把鼠标移动到一个小小的红色方块的过程，你把它当成鼠标走迷宫也可以，总之鼠标要是没有能够准确的移动到下一个预定位置就算失败。如果我们把目标点设置的小一些，那么难度还是很大的。

首先我们给游戏写一个起点，Start：

```
<div id="start">Start</div>
```

至于 CSS，我就把上节课那只红色按钮简单修改了一下，所以不解释喽~

```
#start {
    width: 64px;
    height: 64px;
    padding: 4px;
    position: relative;
    background: #CC0000;
    border: 5px solid #CCCCFF;
    border-radius: 40px;
    font-size: 16px;
    color: #FCFCFC;
    line-height: 64px;
    text-align: center;
}
```



然后我们在刚才的标签里面写一个 div 当作老鼠，于是就变成了：

```
<div id="start">Start
    <div class="mouse"></div>
</div>
```

我再给它一个简单的样式：

```
.mouse {
    display: none;
    width: 10px;
    height: 10px;
    background: #CC0000;
    position: absolute;
    left: 10px;
    top:0;
}
```

写到这里你就发现了，这跟弹出菜单很像啊。对啊，我就是让鼠标放在一个元素上，就会弹出它的内部元素，再放到弹出的元素上，又弹出内部元素……如此玩下去，鼠标一旦位置放错，这些弹出就都隐藏了，于是……游戏失败。

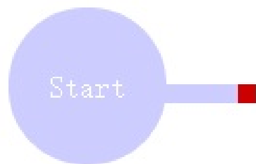
那么显然的我就写下了如下的触发动作：

```
#start:hover {
    background: #CCCCFF;
}
#start>.mouse {
    left: 74px;
    top: 35px;
}
#start:hover>.mouse {
    display: block;
}
.mouse:hover {
    background: #CCCCFF;
}
.mouse:hover>.mouse {
    display: block;
}
```

当然其中包含一些为了视觉舒服而做的小变化。另外就是这个 `#start>.mouse` 是特意规定了一下第一个 .mouse 的位置，因为这个跟别的有点区别。

现在的问题就是 .mouse 里面嵌套 .mouse 了，就像下面这样：

```
<div id="start">Start
    <div class="mouse">
        <div class="mouse">
            <div class="mouse">
                <div class="mouse">
                    <div class="mouse">
                        <div class="mouse">
                            <div class="mouse"></div>
                        </div>
                    </div>
                </div>
            </div>
        </div>
    </div>
</div>
```



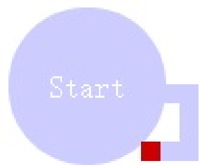
你可以试一下，随着你鼠标的滑动，逐渐显示出了一条直线，因为每一只 mouse 都相对于前一只向后偏移了，所以是一条水平的直线，但是这样没有变化就显得毫无趣味了。那么所谓的变化也就是换个偏移的方向而已，实际上就涉及到了两个属性——left 和 top。

于是我再定义几个类，产生一些变化。

```
.md {
    left: 0;
    top: 10px;
}
.ml {
    left: -10px;
    top: 0;
}
.mu {
    left: 0;
    top: -10px;
}
```

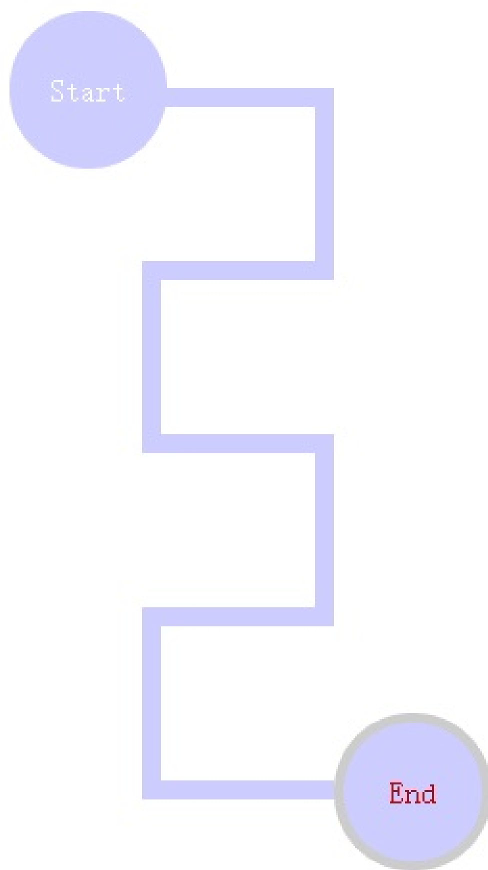
.md 是向下偏移，.ml 向左，.mu 向上，至于向右，.mouse 不就是么？然后这么写：

```
<div id="start">Start
    <div class="mouse">
        <div class="mouse">
            <div class="mouse md">
                <div class="mouse md">
                    <div class="mouse md">
                        <div class="mouse ml">
                            <div class="mouse ml"></div>
                        </div>
                    </div>
                </div>
            </div>
        </div>
    </div>
</div>
```



我就不多写了，你可以自由组合着玩，最后在里面再写一个 #end 来作为结束，就算大功告成。我写了一个案例，你可以下载来玩，当然也要认真想想都是为什么，然后你能想出什么有趣的玩法吗？

我的案例玩起来是这个样子的：



> 本章代码下载：本章代码

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com**（刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发：http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书：http://www.jianshu.com/users/d37eaf3de0ff
- 站酷：http://www.zcool.com.cn/u/12927742

第四十六章 响应式

要是前边的 CSS 部分你学得扎实，这部分基本就是看一眼就懂了，很简单的一个事。就是比今天中午吃啥这个问题还简单。额……我比喻的不恰当哈，那样就成了最难的问题了，算了，不比喻了，你们就领会精神就好了。

先解释一下原理，就是你为一个元素设置多重的 CSS ，他们根据一定的规则，在不同的情况下分别起作用。这一定的条件一般都是指浏览器窗口的宽高范围。

那么我们来做一个小的案例说明一下：

一个 Html 文件，里面一个方块，就是这么简单的内容，这代码大家都能轻松写出来了吧？

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>响应式的小 Box</title>
    <style>
        #box {
            width: 600px;
            height: 300px;
            margin: 50px auto;
            background: blue;
        }
    </style>
</head>
<body>
    <div id="box"></div>
</body>
</html>
```

这个案例简单吧，而且呢，这个 #box 在各种情况下显示的都一样，很好，很老实，但是只是个不聪明的乖宝宝而已。怎么让它变得聪明起来，可以根据外界的变化随机应变呢？当然不是喂他喝脑白金了。

我们改写一下 CSS 部分哦~

```
    <style>
        @media (min-width: 768px){
            #box {
                width: 600px;
                height: 300px;
                margin: 50px auto;
                background: blue;
            }
        }
        @media (max-width: 768px){
            #box {
                width: 90%;
                height: 300px;
                margin: 50px auto;
                background: red;
            }
        }
    </style>
```

看看什么变化？我们来仔细的讲解一下这个代码，其实很好理解的

```
`@media(条件){当满足条件时，要使用的 CSS }`
```

这叫做媒体查询，就是查询一下浏览器窗口的宽度，当然你换成 height，查询高度也没问题。那么我们来说一下上述代码表达的意思：

先做媒体查询，看看当前浏览器的窗口尺寸，如果满足条件 `min-width: 768px` ，就是窗口的最小宽度是 768px ，那么就使用大括号里 CSS ，600px 的宽度，然后蓝色背景。

第二段跟上面类似，就是当窗口的最大宽度是 768px ，那么就执行这段里的 CSS，90% 的宽度，红色背景。

这个效果你要验证的话，要手动拖动窗口的大小去看，别打开看一眼就跑来问我为什么不起作用，那样我不理你哦~

这个其实很简单，只是给某一部分 CSS 加一个条件判断而已，当然了条件可以再复杂一些的，比如加上 and 或者 or 这类的条件。就像下面这样。

```
@media (min-width: 768px) and (max-width: 1024px){
    #box {
        width: 80%;
        height: 300px;
        margin: 50px auto;
        background: green;
    }
}
```

就是表示同时要满足两个条件，大括号里的 CSS 才起作用。如果把 and 换成 or 就是说满足两个条件其中之一就可以了。

这个问题不难，大家试试看就明白了，学代码这东西就是要多尝试，你努力玩他，他就玩不过你了，于是你就赢了，否则……你就被他玩了。

事实上这个媒体查询可以做很多复杂的玩法，有兴趣可以查询一下，祝你看的头晕，怕头晕的先把我说的这些看懂就好，对了 width 也可以换成 height ，什么意思呢？你猜！

然后那个什么 IE6——8，咳咳，算了，不提了，升级 Win10 吧，会更幸福的~

> 本章代码下载：本章代码

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com** （刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发： http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书： http://www.jianshu.com/users/d37eaf3de0ff
- 站酷： http://www.zcool.com.cn/u/12927742

第四十七章 JavaScript Start

那么这节课开始就要学习众神之神的 JavaScript 了。等等，你先别急着吐槽！这个道理是这样的，如果 Html 是平民的话，那么，CSS 简直就是神一样的存在啊，而如此比喻的话，动态的 JavaScript 简直就是众神之神了。这是一个比喻，而且很局限，为此而吐槽的同学，你们的小学语文老师会哭的！

老有人跟我为了这种细节的小问题较劲，真累！当然了其实很多大神都吐槽 JS（JavaScript 的缩写，以后缩写，不打全称了，累）是一款混乱的语言等等……然而其实哪种语言没被如此吐槽过呢？所以，不用在意。然后这次我会尽量把基础的内容给大家介绍一遍，这样你就可以听懂了，然后我会告诉你们如何使用 JQuery ，于是你们就觉得好简单了。嗯，我觉得自己好机智！

好的，那么现在简直就是一个从零开始的新阶段啊。其实学点编程挺好的，无论你是什么行业，什么什么的，反正学了之后你就会发现……原来以前自己的工作方法是多么的原始，原来电脑是应该这么用的！哼哼，跟你们说 QQ 不是右下角弹各种新闻啊，广告啊，不胜其烦嘛，第三方修改版我不是还不放心嘛，TM 或者国际版不是还无法满足我的需求嘛，我就搞了个小工具，自动帮我关弹窗，速度快的我这眼神几乎看不到弹窗，哼，人懒就是这么任性！

反正肯定有有人要吐槽，说什么专注自己行业，两耳不闻窗外事，一生只用 XP 不升级，我不管了，要开讲了，今天废话很多了。

其实 JS 在 Html 文件里的引用方法和 CSS 很像，也是三种方法，所以我一段代码里把这三种都演示一下，反正你以前没见过的部分就是了。

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>JavaScript 引用演示</title>
    <script src="要引用的 js 文件地址"></script>
    <script>
        alert("稻米鼠，你好帅~");
    </script>
</head>
<body>
    <button type="button" onclick="alert('真的，太帅了！')">点击这里查看真相！</button>
</body>
</html>
```

引用外部 js 文件，写在 head 里，还有写在元素里，不讲了不讲了，反正后面反复用呢。先说个简单的东西：

```
alert("这里是要显示的文字");
```

这是一个弹出提示框的命令，我们后面要用它来输出我们操作的结果。要不然没有输出我们怎么知道他干没干活，偷没偷懒？

好了，这节课先到这里，我困了，先睡了。

> 本章代码下载：本章代码

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com**（刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发： http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书： http://www.jianshu.com/users/d37eaf3de0ff
- 站酷： http://www.zcool.com.cn/u/12927742

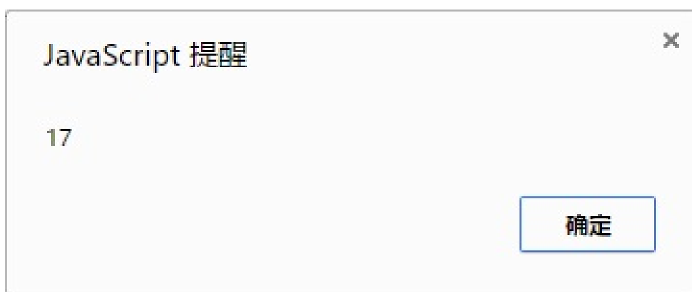
第四十八章 变量

这个......初中，代数，不需要你数学怎么好，基础知识而已。我就不重新去讲了。

那么我们做个最简单的例子来复习一下，超简单的哦~

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>JavaScript 引用演示</title>
    <script>
        a = 12;
        b = 5;
        c = a+b;
        alert(c);
    </script>
</head>
<body>
</body>
</html>
```

那些格式性的代码我们就不说了，直接说 JS 部分。a 这个变量代表 12，b 这个变量代表 5，c 这个变量代表 a 与 b 相加的结果。然后我们让他把 c 的值显示出来，结果就得到了 17。你敢说跟初中的代数，而且是初一的，有什么区别？这也算编程了吧，编写程序让电脑进行计算，你看跟着我三分钟学会编程，成就感爆棚有木有！



这个简单吧？然后呢，a、b、c 这些叫做变量名，就是变量的名字，变量名可以用字母、数字、下划线（_），当然你不能用短划线（-），因为短划线是减号，那你就写成了算式，对吧，这很好理解，然后就是变量必须以字母开头，咳咳，是有特殊情况，但是推荐......听我的！对了，变量名大小写敏感，这是啥意思呢？就是 A 和 a 不是同一个变量。这点小规定不复杂吧，你记住了啊，没记住？没记住就只用小写字母，这样不容易错了吧？

然后你以为变量只可以代表数字么？他还可以代表字符和字符串。这个在初中玩的就很少了哈，其实也挺简单的，看着啊：

```
d = "Hello";
e = "World!";
f = d+ " "+e;
alert(f);
```

那些格式我就不写了，要不然太长了啊。我们要把字符或者字符串放在引号中，单引号双引号都行，别两边不一样就行了，就像你穿一只凉鞋一只皮鞋出去会被人嘲笑的。然后不加双引号你知道那是啥？变量还是数字？是吧，所以一定要用引号标记出来，说明这是字符串。然后字符串居然可以相加，其实相加的结果就是把字符串按照顺序连接在一起。

于是上边的案例就是变量 d 被赋值为 hello，嗯，给变量设定一个值的操作叫做赋值。然后 e 被赋值为了 world! ，然后 f 就被赋值为变量 d 连接上一个空格，再连接上变量 e。然后输出 f ，也挺简单吧。



先记住这些，然后记得 * 是乘号，/ 是除号。() 这是括号，然后你们拿他当计算器玩着吧。

> 本章代码下载：本章代码

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com**（刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发：http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书：http://www.jianshu.com/users/d37eaf3de0ff
- 站酷：http://www.zcool.com.cn/u/12927742

第四十九章 运算符

这几节都是基础课，基础的我一边写一遍打盹。太无聊了，不服气你做二百道一百以内的加减法试试。

上节课我们用了加号，他可以让数字相加，让字符串连接，不解释了啊，那么思考下面的问题哦：

```
heng = "12"+"5";
alert(heng);
```

输出的是什么？自己试试，想想为什么。然后要说一个最基础，最基础的重要问题：js 的每一句都要用分号（;）来结束，嗯哼，依旧有特殊情况，后面讲。

新人最容易犯的错误就是忘记写分号结束这一句，还有就是引号忘记了用英文，没事，我也经常这么范二，习惯就好。

那么基础的运算符上一节结尾都跟大家说了，加减乘除的也太小儿科了，我不讲了，然后括号就是一层套一层，没有什么中括号大括号，但是你的括号要成双配对，不能出现单身狗，这都好理解吧。于是一般的四则运算我们就可以去做了，其实最简单的部分也在上节课演示过了。

然后来说点更简单的，比如我写如下的代码：

```
a = 12;
a = a+5;
alert(a);
```

这个简单吧，a 最后等于 17.那么程序员是懒惰的，嗯，如果这个世界上没有猫咪，也许程序员就是最懒惰的动物了，所以他们觉得， `a = a+5` 这个写法太麻烦了！所以可以简写成： `a += 5` ，这个写法和上面的写法完全一样的效果。这个能理解吧，其实满打满算也就少写了一次变量名，可是我们就是这么懒，你怎么着吧，哼唧~

类似的写法还有 `a -= 5` 、 `a *= 5` 、 `a /= 5` 之类。这个主要是说一下免得你们以后遇到不认识。

然后还有一个很懒惰的写法，类似于 `a = a+1` 这种给变量自己加一的做法在写程序时十分常用，所以我们就要设定一个简单的写法，让自己省力气，于是我们写成 `a++` ，嗯，就是这么简单任性！所以呢， `a--` 就是 `a = a-1` 的意思。

这些并不难，是不是？本来就是，想想游戏里各种兵刃属性什么打孔镶嵌……反正我不会打游戏，听着像天书，我真不知道你们怎么记得那么明明白白，条理清晰的。不过我觉得我说的这些有理有据的，应该比那个好记。你说你是女生不玩游戏？那先粉底还是先防晒霜之类的程序你怎么那么清楚？男生可是怎么也看不懂的。

然后我们继续说啊，刚才那些都是小儿科，学不会是大笨蛋，我们玩点有脑子的东西。初中我们还学过一个东西，就是逻辑性的那个什么什么，我忘记当时叫什么了，反正就是判断一个条件是否成立，成立就为真，不成立就为假，然后，还有真值表，说一些合在一起的结果，什么真真为真，真假为假……我真想不起名字来了，你们回忆起相关知识就好。

别吐槽，我想不起来名字是因为我整天用，但是从来不叫他名字，更何况叫名字也是编程这边的叫法，当年……算了，我承认我老了行不？

那个真假这个事呢在程序里用两个单词表示，ture 和 false。我不会翻译，在我看来这俩单词就是真、失败！然后表示这个意思的变量叫做布尔型，前边有数字的变量，字符串的变量，现在又学一个布尔型。写起来就是：

```
a = ture;
b = false;
```

就是 a 为真， b 为假的意思。那么然后我们就可以往下说一些相关的东西了 —— 比较运算符。

> 大于号，< 小于号，>= 大于等于号，小于等于号 <=。这好理解吧。来写几个算式你看看

```
a = 5;
b = 3;
a > b;
a < b;
a >= b;
a <= b;
```

前两行是给 a、b 赋值，然后 a>b 成立，所以这个比较的结果是 ture，自然地，下一个就是 false，然后 ture，最后一个 false。

你要搞明白的这个比较运算符只是比较这个关系是否成立的，成立就是真，不成立就是假。

然后到一个比较容易混淆的地方了， = 不是等号，他是赋值运算符，他把右侧的值赋给了左侧的变量，你一定要理解这个事情，他完成的是赋值操作，所以你能用他进行比较么？这就好像你能用取款机的取款操作进行查询么？那么比较相等这怎么办？ == 这才是比较是否相等的运算符。写两遍强调一下这个关系啊。这个其实稍微思考一下就能理解的，然后刚才说强调一下的，那么要强调大发劲了是什么？感叹号！有车的车里都带个感叹号的牌子，因为车坏了要用它提示别人，看着点，这有人，你他丫慢点，别撞了！嗯，那么 != 就是不等号。

```
a = 5；
a == 5;
a != 3;
```

这么写一下你就看懂了吧，两个判断都是 ture。

然后所有非零的数字都是 ture ，0 是 false 。这个先记住，后边用就是了。这节课说的不少了，不说了。其实主要是二锅头喝多了，有点晕，你看我这么能扯就明白了。

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com** （刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发： http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书： http://www.jianshu.com/users/d37eaf3de0ff
- 站酷： http://www.zcool.com.cn/u/12927742

第五十章 最简单的逻辑判断

其实我是连续写的，所以还没醒酒，刚才攥着酒瓶子口拧了半天，才发现根本没盖子。

你问我这状态能说明白不？我问你一加一你能立刻反应不？其实不能，一般我们都会愣一下，不是因为太简单了，而是因为这只是我们从来不用，眞尼玛要思考一下。你看我还能做这么复杂的逻辑思维呢，噗，给你开玩笑呢，我写一遍，后期检查一遍，然后规整格式一遍，放心吧你，容易么我。

最简单的逻辑就是 if 判断，这个很像什么呢？就是我们刚讲过去的响应式的媒体查询，媒体查询是当达成了他的条件则执行它内部的 CSS。那么 if 判断就是如果达成小括号里的条件则执行大括号里的语句。

```
if(要判断的条件){
    如果条件为 ture 则执行这部分
}
```

这个其实很好理解，就是如果条件成立，大括号里的与句就在执行，否则就跳过而以。你稍微会点英文就明白，根本就是如果怎样就如何而已。来来，我给大家举个例子：

```
a = 1
if（a==1）{
    alert("a 眞的等于 1 耶！");
}
```

那么因为条件成立，所以语句得以执行。就是这个意思。那么如果你在相反的条件下执行什么的话，你当然可以去写 `if(a!=1){……}` ，但是如果我们需要判断条件之后，如果条件成立则执行某程序，否则则执行另一段程序呢？其实眞的就像我们说英文一样的：

```
a = 2
if（a==1）{
    alert("a 眞的等于 1 耶！");
}else{
    alert("a 竟然不等于 1 啊！");
}
```

如果……否则……就这么简单的逻辑，难吗？？？但是他眞的很难强大！眞的，我不骗你！

然后注意到没有？有几行没有分号耶~因为大括号本身就标志了起始，已经不会引起混乱了，所以最后没有分号。

如果到这里你都看懂了，那么恭喜你，你离入门不远了，现在我给你们留个作业，试着用现在学到的知识去写一个小练习，开发思维哦~看谁写得更加厉害一些了~

> 本章代码下载：本章代码

第五十一章 朕宣你！

首先是一个很郑重地补充：由衷感谢站酷酷友 @璇也 的指正。

事情很简单，在 JS 里面对于变量眞的要求宽泛，像我这种懒人每每都是拿起来就用，然后出问题了才去严格一下他的格式之类。好在我一直写的都是一些简短的小功能，所以就算有性能上的问题，在现在电脑硬件这么强悍的前提下也几乎与我无关。于是……我忘记告诉大家声明变量的事情了。

这是一个很简单的概念，你开始一个工作，要先确定一下员工，点个名，这样大家就知道这些人跟着你干了，然后你再分配他们任务。那么点名这个事情什么时候去做呢？就是你在用这个员工之前点一次就好，整个工程他就都知道跟着你干了。那么怎么声明呢？我写个例子：

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>声明变量</title>
    <script>
        /* 声明一个变量 a */
        var a;
        /* 声明两个变量，b 和 c，当然你可以同时声明更多，记得逗号分隔就行 */
        var b,c;
        /* 声明一个变量 d，同时给他赋値 12 */
        var d = 12;
        /* 声明之后你就可以随心所欲地去使用变量了，跟我们前边就没区别了 */
        a = 6;
        b = 3;
        c = 9;
        alert(a+b+c+d);
    </script>
</head>
<body>

</body>
</html>
```

那么我忘记说了好像也是有好处的，放在现在讲就很容易理解了。你说你不理解？别砸场子好不好，看我后面的案例怎么写的就慢慢理解了，乖，相信我，我是最帅的~

然后是这一章的正文，看标题，看标题，是不是很诱惑？尤其是我还这么帅……

好啦好啦，不与众爱妃说笑了，朕可要说正事了！

上一章我们说了 `if……else……` 语句，就是如果符合条件就怎样，不符合条件又怎样。简单吧，但是呢？非黑即白的判断是不是有点 too simple？这个世界要是果然如此单纯就好了啊！然而经常地我们要面对着更加复杂的选择：如果你考试成绩低于 60 分，你就是个坏学生；如果你的成绩介于 60 分和 80 分之间，你就是个后进生；你的成绩介于 80 分到 95 分之间，你是好学生，你的成绩要是介于 95 到 100 分，那毫无疑问，你是优等生。这个游戏从小学开始老师和家长就每天跟我们玩，大家都很熟悉吧？来看看我怎么用程序表达：

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>选择语句的实例</title>
    <script>
        /* 在案例开始之前，我先说一下注释。
        这是 JS 的多行注释的写法，其实 CSS 里面注释也是这么写的。
        我记不太清我说过没有，但我一定告诉过你，注释不起任何作用，只是为了让我们方便理解，给自己提醒用的。
        那么现在这个注释既然是多行注释，这里面的内容是可以分成几行的 */
        // 这里是 JS 的单行注释，只能写一行
        var a = 98;
        if(a<60){
            alert("你是坏学生~~~")
        }else if(a>=60 && a<80){
            alert("你是后进生~~~")
        }else if(a>=80 && a<95){
            alert("你是好学生~~~")
        }else if(a>=95 && a<=100){
            alert("你是优等生~~~")
        }else if(a>100){
            alert("你是小骗子~~~")
        }
    </script>
</head>
<body>

</body>
</html>
```

那么开始解释，首先说一下 `&&` 符号表示的是 and 的意思，这个我们在前边提到过类似的概念，表示两个条件同时成立才算达成条件。如果你回想起来了初中的那点为数不多的逻辑判断知识就很好理解了。and，就是谁和谁，并肩走，不抛弃不放弃，所以两个条件都是真才算条件成立，有一个为假都不行。

引申出来的 `||` 符号表示 or 的意思，就是或，你或我，只要有一个人去关灯屋里就黑了……额？！好像说错了，当然要是哥俩好，一起去关灯，屋里也黑了……我说不下去了。就是说两个条件有一个为真就算条件达成，当然两个都为真自然也是达成了。

那么顺便把 `!` 说一下，这个是 not，就是否定啊，你说真我就说假。如果结果为真，加上他就是结果为假了。先了解，以后慢慢实践。

回来继续研究我们的逻辑判断，其实跟上节课差不多，都是用 if 进行判断，只是这次连起来了，如果条件成立，那么执行大括号里的语句，否则（else）交给后面的 if 语句继续处理。那么可以看出的一点就是：如果前面已经遇到可以成立的条件，执行了相应的语句之后，后面的那些 if 就不会被判断执行了。认真思考一下这个逻辑顺序。

你不理解我就给你打个比方，我现在是皇上，夜郎国的九五至尊，今晚要睡觉，所以得选个妃子侍寝，但是呢？天天这样的日子，无聊啊，就想了个游戏。我拿了个纸条，上面写上今天要临幸的妃子的名字（就好象上面的 `var a = 98;` ）。然后让妃子们站成一排，从头向尾传纸条，每个妃子拿到纸条要看一眼，如果是自己的名字，那就站出来，否则，把纸条传给下一位。这个游戏都会玩吧？你看，爱妃们的领悟能力都很好嘛~那要是站在第三位的妃子被选中了，她还需要后面的否则吗？当然不用了，所以后面的妃子都没看到纸条，也不用去判断纸条上写的是不是自己了。

那么，爱妃~今夜，朕宣你！

明晚再给大家讲解王公公每次要朕翻牌子的时候都是怎么判断的。

> 本章代码下载：本章代码

上一章 - 目录 - 下一章

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com** （刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发： http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书： http://www.jianshu.com/users/d37eaf3de0ff
- 站酷： http://www.zcool.com.cn/u/12927742

第五十二章 爱妃，朕还宣你！

那个上节课的比喻你可别告诉我女朋友，解释不清，她不会理解夜郎国只有她一位爱妃的……

现在我们先来玩个小游戏，缓解一下你我心中的不安……现在我告诉你一个 1—7 的数字，然后你用这是星期 X 来回答我，就是换个说法，来看看我怎么用程序表达：

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>选择语句的实例</title>
    <script>
        var a = 5;
        switch(a)
        {
            case 1:
                alert("星期一");
                break;
            case 2:
                alert("星期二");
                break;
            case 3:
                alert("星期三");
                break;
            case 4:
                alert("星期四");
                break;
            case 5:
                alert("星期五");
                break;
            case 6:
                alert("星期六");
                break;
            default:
                alert("星期日");
        }
    </script>
</head>
<body>

</body>
</html>
```

这是一个 switch 语句，他重在选择，而前边学的 if 语句重在判断，咳，这是我自己总结的。

switch 后面的括号里写上他的条件，然后我们为他的条件求值，然后拿着这个值去大括号里跟 case 后面的值进行比较，如果这两个值相等了，就执行冒号后面的语句，我们这里写的都是弹出提示，这个你看的懂了。然后有一个 `break;` ，这个要解释一下，这是打破当前的流程，你都找到对应的值了，没必要再往下看了，所以到此打住，别浪费时间往下看了。最后还有一个 `default` ，这是说如果上面的情况都没法满足，那就执行这个吧。默认的，不得已的最后选择，所以他写在最后。因为他后面也没啥了，所以也就不用 `break;` 了。

那么我们现在 a=5，所以拿着这个值（5）去下面比较， `case 1:` 这个值是 1 ，跟 5 不相等，跳过，看下一个，2、3、4 都是不相等的，到了 `case 5:` ，这个条件符合了，所以弹出“星期五”，然后后面的就不看了。

理解了吧？再给你举个例子，非得说这么XX的段子你才能记住，真不怕我跪搓板么？我现在有九万六千个妃子（刚说好的只有一个呢......），人太多，记不住了，都编号了。王公公抱个花名册，上面写着：王贵妃 1号；李贵妃 2号；......然后我翻牌子，是一万七千九百二十八号。王公公赶紧抱着花名册查啊，一看，王贵妃是 1 号，跟我选的数字不符，跳过，李贵妃......跳过......一直到田园二狗这里，一看，正好是一万七千九百二十八号，于是赶紧传令下去，让田园二狗准备着，至于后面那将近八万号就不再看了。要是没查到对应的号，皇上就直接去默认的皇后那里去就对了。

你看这么一说你就懂了吧？猥琐的口水都出来了，我估计现在跪搓板已经解决不了问题了，我得抱着田园二狗在外面睡俩月了。

> 本章代码下载：本章代码

上一章 - 目录 - 下一章

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com** （刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发： http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书： http://www.jianshu.com/users/d37eaf3de0ff
- 站酷： http://www.zcool.com.cn/u/12927742

第五十三章 一百遍啊一百遍！

这不前边例子举得太YY，被女盆友抓住了么，让我说一百遍我错了。这么轻的处罚，是多么的同情达理啊。

重复的事情程序员从来都不会重复去做的，因为我们有循环。来我们一起写一个糊弄女盆友。

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>一百遍我错了！</title>
    <script>
        for (var i = 0; i < 100; i++) {
            alert("我错了~第 "+i+"遍");
        };
    </script>
</head>
<body>

</body>
</html>
```

那么解释一下子如此简单的代码：for 就是按着括号里的条件，循环执行大括号里的语句。这里比较复杂的是小括号里的条件，它分为三句话，当然这不是三段论啊。第一句设定一个变量，给他一个初始值；第二句，如果这个变量符合这句的条件，那么执行大括号里的内容；第三句，当大括号里的内容执行完了，执行第三句，然后重新判断第二句的条件，如果符合则执行大括号里的内容……如此往复不断。

你要是实验这个程序，一定要记得别写 100 ，写个 5 就差不多了，要不然你就要不断地关闭提示框，一百遍啊，一百遍！

再打个比方，规规矩矩的。说一百遍同时还要数着次数，这不方便啊，所以我面前摆了个碗，空的，我看一眼碗里的豆子不足 100 粒，于是大声说："我错了！"然后往碗里扔一粒豆子，看看依旧不足 100 粒，再说，再扔……直到碗里的豆子是一百粒为止。这个道理是完全一样的。

初始值你可以给其他的值，但是程序员一般喜欢从零开始数，这个你要理解，否则很容易搞出错误来。然后 i++ 这个也可以换做其他的，那么我们再举一个例子，让我们数出从零到一百的所有偶数吧：

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>从零到一百的所有偶数</title>
    <script>
        for (var i = 0; i <= 100; i=i+2) {
            alert(i);
        };
    </script>
</head>
<body>

</body>
</html>
```

来看看，我改了什么？条件改为 i<=100，这样等于一百的时候依旧满足条件，再执行一次程序，就把 100 也报出来了，后面 i=i+2，其实就是一次往碗里扔两颗豆子。

> 本章代码下载：本章代码

上一章 - 目录 - 下一章

第五十四章 一万遍啊一万遍！

我觉得咱们里面有叛徒，谁又跟我女盆友说我作弊的事了？现在她罚我说我错了一万遍，而且不许使用 for 循环。

可是……我们还有 while 循环呀！这个其实跟 for 循环很像的，我写得出来你猜猜意思。

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>while 循环</title>
    <script>
        var i=0;
        while(i<10000){
            alert("我错了~第 "+i+"遍");
            i++;
        }
    </script>
</head>
<body>

</body>
</html>
```

对照着 for 循环你很容易理解吧，先判断小括号里条件是否成立，然后如果成立就执行大括号里的语句，执行完再判断小括号里的条件是否成立，如果成立再执行大括号里……

你看我完成了女朋友的苛刻条件了吧？但是作为一个优秀的男盆友怎么可以就此止步呢？所以事实上我写的程序更加的简洁：

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>while 循环</title>
    <script>
        while(1){
            alert("我错了~");
        }
    </script>
</head>
<body>

</body>
</html>
```

我们说过非零的数字为 true，条件成立，然后执行了大括号的语句之后，小括号里的条件依旧成立，其实这个条件永远不会发生变化，这是一个常量（常量不会变），所以条件一直成立，循环一直继续，这就是传说中可以跑到地老天荒的死循环，我们写程序的时候一定要避免写死循环。

现在女盆友还在那里一边计数一边关提示框呢。

> 本章代码下载：本章代码

上一章 - 目录 - 下一章

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com**（刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发： http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书： http://www.jianshu.com/users/d37eaf3de0ff
- 站酷： http://www.zcool.com.cn/u/12927742

第五十五章 苛刻至极！

女盆友终于发现了问题，那时候她已经数到了三万五千四百二十三次……

她没有发脾气，只是让我写一个条件不成立的循环，但是还要这个循环完成一次动作。这分明就是又要马儿跑得快，又要马儿不吃草啊！

怕你们不明白，我解释一下，这分明就是既不给我喝咖啡，又要我勤更新……

不闹了，但是你要往心里去啊！

完成女盆友的苛刻任务，谁让我爱她爱的那么深。

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>奇特的循环</title>
    <script>
        var a=1;
        do {
            alert("我成功了");
        }while(a==2);
    </script>
</head>
<body>

</body>
</html>
```

这是 while 循环的一个变种，do...while...，他是一个愣头青式的循环，无论如何都要先把大括号里的语句执行一遍，然后再去看小括号里的条件， 如果条件成立则执行下一遍。所以我写的个代码里虽然条件不成立，但是大括号里的语句还是被执行了一遍，嘿嘿嘿……我得救了。

JS 的内容我打算就说这么多了，虽然还有很多很多东西值得去学习，我不管了，这里带你们入门了，你们自学起来难度也应该不是很大了。然后我们下一节开始讲 JQuery，因为这简直是前端新人的神器，简单而强大。第一次认识 JQ 的时候我差点感动哭了，咋就能这么简单好用呢？

最近有些事情要忙，这个系列下次更新可能需要等一小段时间，但是温暖的咖啡也许可以召唤他早日归来。

视频的手把手教程也已经开始了，有兴趣的加群询问报名吧。

然后最近有零碎时间，我可能在站酷发一些“我就聊聊”系列，分享点日常的感想和技巧。

> 本章代码下载：本章代码

上一章 - 目录 - 下一章

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com**（刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发：http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书：http://www.jianshu.com/users/d37eaf3de0ff
- 站酷：http://www.zcool.com.cn/u/12927742

第五十六章 一套工具

JavaScript 我们讲了一些了，但是都是些基础，甚至你会觉得有些飘渺。因为我们说学前端，可是现在没看出 JS 与前端有很大的关系。为什么呢？因为那些关系紧密的地方我都没讲啊！

不是不想讲啊，关键是我也不会啊……（众人“噫”，退场中……）

可是我的话还没说完啊！我，我不会怎么混到今天的啊！所以，我是眞不会……先别走啊！但是我有简单的办法啊！我靠，转了一大圈，总算说到点子上了。难度

这个方法就是——JQuery。不得不说，JS 我眞的压根就没认眞看过几眼，前面那些循环啊什么的基础我都是从其它语言中学的，到 JS 里也就是语法细微的变化一下。所以前边的基础你学习别的语言的时候一样有用，别忽视哦！至于后面深层次的东西我还没学就遇到了 JQuery，然后就再也没然后了。

说的这么热闹，那么 JQuery 究竟是个什么东西呢？我来打个比方您就明白了。

假如有了铁，我们知道我们可以用它创造许许多多的事物，但是，这个难度并不是谁都可以接受的。但是，如果我给你一套工具（锤子、扳手……）以及足够的零件，那情况可就不一样了。虽然依旧是铁，但是只要不是太娇弱的女生，基本都可以用这些东西做点什么了。

JQuery 呢，就是用 JS 这块铁，打造出来的一套工具。本质量铁，但是更好用。

道理就是介么个道理，但是说的时候我们说：JQuery 是 JavaScript 的一个库。你可以理解为 JQuery 是基于 JS 建立的一套工具集，两者可以很好的配合使用，就好像用锤子打铁那样的热火朝天，亲密无间。而且说到一个，也就说明这只是其中一个，还有好多好多这样的库（工具集）我们还没有提到。

好，解决了它是什么的问题，我们再来说一说它能做什么。我个人总结，JQuery 可以让我们十分简单、方便、自由的操纵页面中的元素。这就好像什么呢？我是总经理，手底下有许多的员工，但我并不善于事无巨细的去指挥他们。所以我找了一个助理，这个助理非常善于指挥，现在我有什么事只需要跟他（女盆友说必须是单人旁）交待一下就好，倍儿简单，倍儿轻松！有了这么好用的人才，我自然是懒得再自己费力去学指挥人的技巧了，所以我至今也没认眞学过 JS。

当然，对于新手，我一直强调——先求效果，再讲道理。否则没有趣味性，没有成就感，很容易学不下去的。所以，先学先用，逐步深入也没什么问题。

然后的问题就是怎么去用了。这个……下回分解！

上一章 - 目录 - 下一章

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com** （刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发： http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书： http://www.jianshu.com/users/d37eaf3de0ff
- 站酷： http://www.zcool.com.cn/u/12927742

第五十七章 助理！麻烦过来一下~

助理妹子并没有过来，因为……我并没有助理啊！

上一课说到怎么用，那么到底怎么用呢？嗯，就是像标题那么用啦~

你使用工具去创造，总要先把工具箱拎过来才好开工；经理依靠助理管理团队，总要先把助理叫过来才好交待任务。我们也是一样，要先把 JQuery 叫过来才能使用，叫它过来这个动作称作“引入”。

引入 JQuery 非常简单，和引入一般的 JS 无异。所以常用的有两种方式：本地引用和 CDN 引用。

本地引用呢，自然是将 JQuery 下载到本地，和其它的网页文件放在一起，然后引用它。那么现在的问题就是在哪里下载它了。

下载东西最好的去处就是它的官网，所以我们要到 JQuery 的官网（https://jquery.com/）进行下载。但是很多同学一点下“Download”的按钮就懵了，因为并没有立刻开始下载，而是跳转到了一个下载页面，里面有好多链接，完全不知道接下来点击哪一个好。在这里我来给大家解释两个最常用的链接。

`Download the compressed, production jQuery 3.0.0` ——“compressed”是已经被压缩的，就是去掉了非必须的空白和换行以减少文件体积；“production” 意思是产品吧，既然已经是成品了，自然是可以拿来就用的，这也是我们日常主要使用的版本。换言之：这是用在生产环境的版本。

`Download the uncompressed, development jQuery 3.0.0` ——“uncompressed”没有进行压缩的，就是保留着清晰缩进、换行和注释的，，当然体积也比较大；“development”开发，所以，这个版本是用来阅读、学习和二次开发的。

本来这也就够了，但是由于一些历史原因，我们还不得不说说这几个版本号。1.X （就是由 1. 开头的版本，下同）版，是早期版本，所以兼容较低版本的浏览器；随着发展，进入了 2.X 时代，显著特点就是不兼容低版本浏览器；最近又出了 3.X，反正又排除了一些旧的东西。据此，我们就可以选择我们要使用的版本了。我们这里选用 3.0.0 版。

于是，我们的引入代码就类似于酱紫——

```
<script type="text/javascript" src="jquery-3.0.0.min.js"></script>
```

当然，相对地址的正确性要由你根据实际情况去进行修改。

然后说 CDN 引入法，不过在这之前，我们得先说说啥是 CDN。CDN，全称是 Content Delivery Network，翻译过来是内容分发网络。然而并听不懂！其实简言之，CDN 和京东家的库房是一个意思的。

比如本宝宝的书上市了，京东独家首发。那是相当的火爆啊，那家伙，那眞是锣鼓喧天......我靠！你们别扔皮鞋！！！

京东可是次日（当日）送达啊，现在全国各地都在抢购，如果他们坚持每一本书都从出版社拿货，再送到大家手里，那就是眞的累成了狗，也未必就能按时送达，这样大家就不开心了，甚至索性不买了，那本宝宝肯定就要不开薰了！

但是，京东并还是这样做的，所以我也并不需要急着哭泣。事实上京东在全国各地有许多的库房，他们会预先将这些可能在全国火爆销售的产品储存在各个库房（为什么说到这一句我觉得这么美！）。然后哈尔滨的朋友购买就直接从黑龙江的库房发货，海南的朋友购买就从广东发货......这样效率就高多了。

CDN 也是一样，我们预先把网站中的资源存放在各地的服务器中，在需要的时候，由就近的服务器将资源发送给读者，以提高访问速度。

那么像 JQuery 这么常用的库，自然已经有很多 CDN 服务存放了，我们只要选择一个，进行引用，就可以直接享受这份便利了。比如：

```
<script type="text/javascript" src="//cdn.bootcss.com/jquery/3.0.0/jquery.min.js"></script>
```

然后，我们注意到这个地址和我们以往见到的地址有所不同，`//` 的前面并没有 `http:` 或者 `https:`，其实这样写可以同时兼容这两种协议。但是，大家在本地测试的时候，这样写可能就没法有效的引用了，所以，在必要时可以自己添加 `http:` 以便可以在本地进行测试。

上一章 - 目录 - 下一章

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com**（刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发：http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书：http://www.jianshu.com/users/d37eaf3de0ff
- 站酷：http://www.zcool.com.cn/u/12927742

第五十八章 各就各位，预备~~

助理来了，我要交待任务了。然而程序是愚蠢的，它只能按部就班的去完成我交待的任务，再多一点变通都不会，哎呀~捉鸡！

所以我们必须按着它的语言习惯，事无巨细的跟它交待清楚一切，这就是语法和算法。那么 JQuery 的语法是怎样的呢？

它完全兼容 JavaScript 的语法（废话，它就是 JS 啊），然后它自己的语法粗略的看，可以认为只有一句：

> 谁 在什么时候/它的什么属性 怎么样

请务必将这句话背诵下来，据说可以强身健体，益寿延年，纯属扯淡……但是它对于你理解 JQuery 却会有神奇的帮助。

好了，这个话题暂且放下，我们想想要交待它做什么事情吧，比如张三去干啥，李四去干啥之类。可是我要真这么说了肯定出事。因为我丢掉了一个前提：等人来了之后！傻叉助理接到这个没有前提的指令之后，马不停蹄的去执行，可是人们还没来啊，于是它就只好转回来告诉我：它找不到要把命令传达到的对象。这就是传说中的“单身狗 ”！额，其实叫作“找不到对象”。

所以，一般在写 JQ 的时候我们都会先强调一下，要等所有人都到齐之后再开始。那么我们来看看这个每次都要交待的前提应当如何去写。

```
$(document).ready(function(){
    ……
});
```

我们逐一解释，$ 是一个标记，说明这里是 JQ；它后面的小括号里面是选择器，用来选择要执行该任务的对象，这里的选择器和 CSS 的选择器很像，但更加强大。一般的，选择器要用引号标记起来，当做一个字符串来处理。但是特殊的选择器则不用引号，因为这些预先设定好的选择器是被当作常量的。这里的 `document` 便是一个，表示选择的是整个文档。 `.` 是一个分割，或者说是转折，你可以先简单的将它理解为“的”，或者“当……时候”； `.` 后面是属性、事件、方法等等。之后的括号里是参数和要执行的代码之类。

所以上面一句的意思是：当 文档 准备好之后 执行 `function` （函数）。换言之，就是等人都到齐了再开始分配任务，开展工作。

上一章 - 目录 - 下一章

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com** （刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发： http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书： http://www.jianshu.com/users/d37eaf3de0ff
- 站酷： http://www.zcool.com.cn/u/12927742

第五十九章 见证奇迹

说了这么多，该动动手了，其实很简单的。

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>JQuery 的第一个测试</title>
    <script type="text/javascript" src="http://cdn.bootcss.com/jquery/3.0.0/jquery.min.js"></script>
    <script>
        $(document).ready(function(){
            $("button").click(function(){
                alert("就这样被你征服~~！");
            });
        });
    </script>
</head>
<body>
    <button>点我试一下</button>
</body>
</html>
```

我就解释一点东东， `function(){}` 这个是一个函数，在这里的作用是让大括号里的语句形成一个整体，然后以这个整体来作为这个 JQ 语句的一个参数。

然后再进一步， `click` 是单击的意思。仔细读一读，试一试，想一想，是不是觉得可以看懂了？就是这么简单啊，难么？

`dblclick` 是双击，改一改，看看你想的对不对。

本节课过于简单，老师拒绝授课，并趴在讲桌上开始睡觉。

上一章 - 目录 - 下一章

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> alay9999@163.com （刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发： http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/
- 简书： http://www.jianshu.com/users/d37eaf3de0ff
- 站酷： http://www.zcool.com.cn/u/12927742

第六十章 开始玩玩

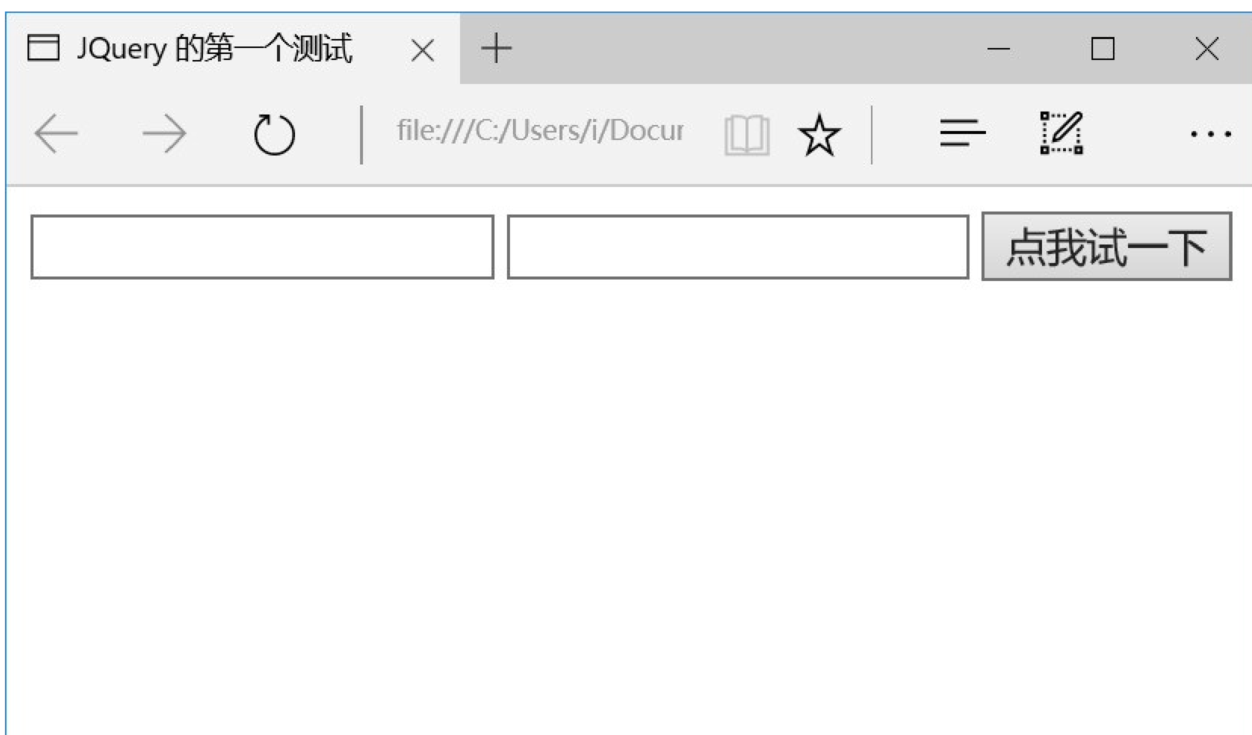
当发生某个特定事件的时候，程序会做出相应的反应，那么就是程序对这一事件进行了响应。我们也说我们把某些代码绑定到了这一事件上。 那么现在我们就认识了单击和双击事件，当然还有很多其它的事件，但是因为用法基本一样，比较简单，我们就不再一一敷述，大家自行查看手册即可。

当然，其实在后面也是会不断涉及的。

这不好玩！但是，再加上属性就有点意思了。其实真写起来你未必会在乎哪个是事件，哪个是属性的。来看我~~七十二变~~写个~~栗子~~例子。

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>JQuery 的第一个测试</title>
    <script type="text/javascript" src="http://cdn.bootcss.com/jquery/3.0.0/jquery.min.js"></script>
    <script>
        $(document).ready(function(){
            $("button").click(function(){
                var text = $("#one").val();
                $("#two").val(text);
            });
        });
    </script>
</head>
<body>
    <input type="text" id="one">
    <input type="text" id="two">
    <button>点我试一下</button>
</body>
</html>
```

试试这个吧~我来教大家怎么玩。



打开之后有两个输入框和一个按钮，先在第一个输入框内输入任意文字，然后点击按钮试试看~

你会发现第一个输入框的内容被复制到了第二个输入框里。那么究竟发生了什么呢，我们一起来分析一下代码：

当按钮惨遭单击之后，依次发生了两件事情。第一件，获得当前 `#one` 元素的值，并存入变量 `text` 中。第二件，将 `#two` 元素的值，设置为变量 `text` 的值。

所以， `val` 可以做两件事情，获取和设置。如果小括号里面是空的，则获取这个元素的值；否则，将小括号里的值设置给这个元素。

事情从此变得有意思起来了呢！

上一章 - 目录 - 下一章

本书是收费的，不过交费凭自觉。价格定义为每人请我喝一杯咖啡（哪种品质的咖啡随意），支付宝账号：

> **alay9999@163.com**（刘源）

为了让大家阅读方便，本书将在如下站点发布，但最终内容以主站为准：

- 主站首发： http://coffee.zji.me/
- Gitbook: http://codeme.gitbooks.io/easy-web-code-book/

- 简书： http://www.jianshu.com/users/d37eaf3de0ff
- 站酷： http://www.zcool.com.cn/u/12927742